Lecture Notes in Computer Science 4448

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Editorial Board

David Hutchison
Lancaster University, UK

Takeo Kanade
Carnegie Mellon University, Pittsburgh, PA, USA

Josef Kittler
University of Surrey, Guildford, UK

Jon M. Kleinberg
Cornell University, Ithaca, NY, USA

Friedemann Mattern
ETH Zurich, Switzerland

John C. Mitchell
Stanford University, CA, USA

Moni Naor
Weizmann Institute of Science, Rehovot, Israel

Oscar Nierstrasz
University of Bern, Switzerland

C. Pandu Rangan
Indian Institute of Technology, Madras, India

Bernhard Steffen
University of Dortmund, Germany

Madhu Sudan
Massachusetts Institute of Technology, MA, USA

Demetri Terzopoulos
University of California, Los Angeles, CA, USA

Doug Tygar
University of California, Berkeley, CA, USA

Moshe Y. Vardi
Rice University, Houston, TX, USA

Gerhard Weikum
Max-Planck Institute of Computer Science, Saarbruecken, Germany

Mario Giacobini et al. (Eds.)

Applications of Evolutionary Computing

EvoWorkshops 2007: EvoCOMNET, EvoFIN, EvoIASP, EvoINTERACTION, EvoMUSART, EvoSTOC and EvoTRANSLOG
Valencia, Spain, April 11-13, 2007
Proceedings

Springer

Volume Editors

see next page

Cover illustration: Morphogenesis series #12 by Jon McCormack, 2006

Library of Congress Control Number: 2007923848

CR Subject Classification (1998): F.1, D.1, B, C.2, J.3, I.4, J.5

LNCS Sublibrary: SL 1 – Theoretical Computer Science and General Issues

ISSN 0302-9743
ISBN-10 3-540-71804-4 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-71804-8 Springer Berlin Heidelberg New York

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, re-use of illustrations, recitation, broadcasting, reproduction on microfilms or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer. Violations are liable to prosecution under the German Copyright Law.

Springer is a part of Springer Science+Business Media

springer.com

© Springer-Verlag Berlin Heidelberg 2007
Printed in Germany

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 12044283 06/3180 5 4 3 2 1 0

Volume Editors

Mario Giacobini
Dept. of Animal Production,
Epidemiology and Ecology
University of Torino, Italy
mario.giacobini@unito.it

Anthony Brabazon
School of Business
University College Dublin, Ireland
anthony.brabazon@ucd.ie

Stefano Cagnoni
Dept. of Computer Engineering
University of Parma, Italy
cagnoni@ce.unipr.it

Gianni A. Di Caro
IDSIA, Switzerland
gianni@idsia.ch

Rolf Drechsler
Institute of Computer Science
University of Bremen, Germany
drechsle@informatik.uni-bremen.de

Muddassar Farooq
Center for Advanced Studies in
Engineering, Pakistan
muddassar.farooq@case.edu.pk

Andreas Fink
Fac. of Economics & Social Sciences
Helmut-Schmidt-University
Hamburg, Germany
andreas.fink@hsu-hamburg.de

Evelyne Lutton
INRIA Rocquencourt, France
Evelyne.Lutton@inria.fr

Penousal Machado
Dept. of Informatics Engineering
University of Coimbra, Portugal
machado@dei.uc.pt

Stefan Minner
Dept. of Logistics
University of Mannheim, Germany
minner@bwl.uni-mannheim.de

Michael O'Neill
School of Computer Science and
Informatics
University College Dublin, Ireland
m.oneill@ucd.ie

Juan Romero
Facultad de Informatica
University of A Coruña, Spain
jj@udc.es

Franz Rothlauf
Dept. of Information Systems
Johannes Gutenberg University
Mainz, Germany
rothlauf@uni-mainz.de

Giovanni Squillero
Dip. di Automatica e Informatica
Politecnico di Torino, Italy
giovanni.squillero@polito.it

Hideyuki Takagi
Faculty of Design
Kyushu University, Japan
takagi@design.kyushu-u.ac.jp

A. Şima Uyar
Dept. of Computer Engineering
Istanbul Technical University, Turkey
etaner@cs.itu.edu.tr

Shengxiang Yang
Dept. of Computer Science
University of Leicester, UK
s.yang@mcs.le.ac.uk

Preface

Evolutionary computation (EC) techniques are efficient nature-inspired planning and optimization methods based on the principles of natural evolution and genetics. Due to their efficiency and simple underlying principles, these methods can be used in the context of problem solving, optimization, and machine learning. A large and continuously increasing number of researchers and professionals make use of EC techniques in various application domains. This volume presents a careful selection of relevant EC examples combined with a thorough examination of the techniques used in EC. These papers illustrate the current state of the art in the application of EC and should help and inspire researchers and professionals to develop efficient EC methods for design and problem solving.

All papers in this book were presented during EvoWorkshops 2007, which was a varying collection of workshops on application-oriented aspects of EC. The year 2007 was especially important for EvoWorkshops, which celebrated their tenth edition. In fact, since 1998, EvoWorkshops has provided a unique opportunity for EC researchers to meet and discuss application aspects of EC and it has been an important link between EC research and its application in a variety of domains. During these ten years, new workshops have arisen, some have disappeared, while others have matured to become conferences of their own, such as EuroGP in 2000, EvoCOP in 2004, and EvoBIO in 2007.

Another fundamental novelty in 2007 was the creation of EVO*, Europe's premier co-located events in the field of evolutionary computing, unifying EuroGP, the main European event dedicated to genetic programming, EvoCOP, the main European conference on evolutionary computation in combinatorial optimization, EvoBIO, the first European conference on EC and related techniques in bioinformatics and computational biology, and EvoWorkshops. The proceedings for all of these events, EuroGP 2007, EvoCOP 2007 and EvoBIO 2007, are also available in the LNCS series (number 4445, 4446, and 4447). EVO* was held in Valencia, Spain, during April 11–13, 2007, jointly with the conferences EuroGP 2007, EvoCOP 2007, and EvoBIO 2007.

EvoWorkshops 2007 consisted of the following individual workshops:

- *EvoCOMNET*, the Fourth European Workshop on the Application of Nature-Inspired Techniques to Telecommunication Networks and Other Connected Systems
- *EvoFIN*, the 1st European Workshop on Evolutionary Computation in Finance and Economics
- *EvoIASP*, the Ninth European Workshop on Evolutionary Computation in Image Analysis and Signal Processing
- *EvoInteraction*, the 2nd European Workshop on Interactive Evolution and Humanized Computational Intelligence

- *EvoMUSART*, the Fifth European Workshop on Evolutionary Music and Art
- *EvoSTOC*, the Fourth European Workshop on Evolutionary Algorithms in Stochastic and Dynamic Environments
- *EvoTransLog*, the 1st European Workshop on Evolutionary Computation in Transportation and Logistics

EvoCOMNET addresses the application of EC techniques to problems in communications, networks, and connected systems. New communication technologies, the creation of interconnected communication and information networks such as the Internet, new types of interpersonal and interorganizational communication, and the integration and interconnection of production centers and industries are the driving forces on the road towards a connected, networked society. EC techniques are important tools to be able to face these challenges.

EvoFIN is the first European event specifically dedicated to the applications of EC and related natural computing methodologies, to finance and economics. Financial environments are typically hard, being dynamic, high-dimensional, noisy and co-evolutionary. These environments serve as an interesting test bed for novel evolutionary methodologies. The papers for this year's workshop encompassed several key topics in finance including portfolio optimization, time series forecasting, risk management, failure prediction, agent behavior, and options pricing.

EvoIASP, the longest-running of all EvoWorkshops which reached its ninth edition in 2007, has been the first international event solely dedicated to the applications of EC to image analysis and signal processing in complex domains of high industrial and social relevance.

EvoInteraction deals with various aspects of interactive evolution, and more broadly of computational intelligence in interaction with human intelligence, including methodology, theoretical issues, and new applications. Interaction with humans raises several problems, mainly linked to what has been called the user bottleneck, i.e., human fatigue.

EvoMUSART focuses on the use of EC techniques for the development of creative systems. There is a growing interest in the application of these techniques in fields such as art, music, architecture, and design. The goal of EvoMUSART is to bring together researchers that use EC in this context, providing an opportunity to promote, present, and discuss the latest work in the area, fostering its further developments and collaboration among researchers.

EvoSTOC addresses the application of EC in stochastic environments. This includes optimization problems with changing, noisy, and/or approximated fitness functions and optimization problems that require robust solutions. These topics recently gained increasing attention in the EC community and EvoSTOC was the first workshop that provided a platform to present and discuss the latest research in this field.

EvoTransLog deals with all aspects of the use of evolutionary computation, local search and other nature-inspired optimization and design techniques for the transportation and logistics domain. The impact of transportation and logistics

on the modern economy and society has been growing steadily over the last few decades. Along with the development of more powerful computer systems, design and optimization techniques such as evolutionary computing approaches have been developed allowing one to use computer systems for systematic design, optimization, and improvement of systems in the transportation and logistics domain.

EvoWorkshops 2007 continued the tradition of providing researchers, as well as people from industry, students, and interested newcomers, with an opportunity to present new results, discuss current developments and applications, or simply become acquainted with the world of EC. Moreover, it encourages and reinforces future possible synergies and interactions between members of all scientific communities that may benefit from EC techniques.

This year, the EvoWorkshops had the highest number of submissions ever, cumulating at 160 entries (with respect to 143 in 2005 and 149 in 2006). EvoWorkshops 2007 accepted ten-page full papers and eight-page short papers. Full papers were presented orally over the three conference days, while short papers were presented and discussed during a special poster session. The low acceptance rate of 34.37% for EvoWorkshops 2007, the lowest of all past editions, along with the significant and still-growing number of submissions is an indicator of the high quality of the articles presented at the workshops, showing the liveliness of the scientific movement in the corresponding fields. The following table shows relevant statistics for EvoWorkshops 2006 and EvoWorkshops 2007 (accepted short papers are in brackets):

Year	2007			2006		
	Submissions	Accept	Ratio	Submissions	Accept	Ratio
EvoCOMNET	44	11(7)	25%	16	5	31.2%
EvoFIN	13	6(2)	46.15%	-	-	-
EvoIASP	35	11(10)	31.43%	35	12(7)	34.3%
EvoInteraction	7	4	57.14%	8	6	75%
EvoMUSART	30	10(5)	33.33%	29	10(4)	34.5%
EvoSTOC	11	5	45.45%	12	6(2)	50.0%
EvoTransLog	20	8	40%	-	-	-
Total	160	55(24)	34.37%	149	65(13)	43.6%

We would like to thank the following institutions:

- The Universidad Politécnica de Valencia, for its institutional and financial support and for providing premises and administrative assistance
- The Instituto Tecnológico de Informatica in Valencia, for cooperation and help with local arrangements
- The Spanish Ministerio de Educación y Ciencia, for their financial contribution
- The Centre for Emergent Computing at Napier University in Edinburgh, Scotland, for administrative help and event coordination

Even with an excellent support and location, an event like EVO* would not have been feasible without authors submitting their work, members of the Program Committees dedicating energy in reviewing those papers, and an audience. All these people deserve our gratitude.

Finally, we are grateful to all those involved in the preparation of the event, especially Jennifer Willies for her unaltered dedication to the coordination of the event over the years. Without her support, running such a type of conference with a large number of different organizers and different opinions would be unmanageable. Further thanks to the local organizer Anna I. Esparcia-Alcázar, Ken Sharman, and the Complex Adaptive Systems group of the Instituto Tecnológico de Informàtica for making the organization of such an event possible in a place as unique as Valencia. Last but surely not least, we want to especially acknowledge Leonardo Vanneschi for his hard work as Publicity Chair of the event, and Marc Schoenauer for his continuous help in setting up and maintaining the conference portal.

April 2007

Mario Giacobini
Gianni A. Di Caro
Andreas Fink
Stefan Minner
Franz Rothlauf
A. Şima Uyar
Anthony Brabazon
Rolf Drechsler
Evelyne Lutton
Michael O'Neill
Giovanni Squillero
Shengxiang Yang
Stefano Cagnoni
Muddassar Farooq
Penousal Machado
Juan Romero
Hideyuki Takagi

Organization

EvoWorkshops 2007 was part of EVO* 2007, Europe's premier co-located events in the field of evolutionary computing, that also included the conferences EuroGP 2007, EvoCOP 2007, and EvoBIO 2007.

Organizing Committee

EvoWorkshops Chair	Mario Giacobini, University of Torino, Italy
Local Chair	Anna Isabel Esparcia-Alcazar, Universdad Politécnica de Valencia, Spain
Publicity Chair	Leonardo Vanneschi, University of Milano Bicocca, Italy
EvoCOMNET Co-chairs	Muddassar Farooq, Center for Advanced Studies in Engineering, Pakistan Gianni A. Di Caro, IDSIA, Switzerland
EvoFIN Co-chairs	Anthony Brabazon, University College Dublin, Ireland Michael O'Neill, University College Dublin, Ireland
EvoIASP Chair	Stefano Cagnoni, University of Parma, Italy
EvoInteraction Co-chairs	Evelyne Lutton, INRIA, France Hideyuki Takagi, Kyushu University, Japan
EvoMUSART Co-chairs	Juan Romero, University of A Coruña, Spain, Penousal Machado, University of Coimbra, Portugal
EvoSTOC Co-chairs	A. Şima Uyar, Istanbul Technical University, Turkey Shengxiang Yang, University of Leicester, UK
EvoTransLog Co-chairs	Andreas Fink, Helmut-Schmidt-University Hamburg, Germany Stefan Minner, University of Mannheim, Germany Franz Rothlauf, Johannes Gutenberg University Mainz, Germany

Program Committees

EvoCOMNET Program Committee

Payman Arabshahi, University of Washington, USA
Eric Bonabeau, Icosystem Corp., USA
Frederick Ducatelle, IDSIA, Switzerland
Luca M. Gambardella, IDSIA, Switzerland

Jin-Kao Hao, University of Angers, France
Marc Heissenbuettel, Swisscom Mobile Ltd., Switzerland
Malcolm I. Heywood, Dalhousie University, Canada
Nur Zincir-Heywood, Dalhousie University, Canada
Bryant Julstrom, St. Cloud State University, USA
Vittorio Maniezzo, University of Bologna, Italy
Alcherio Martinoli, EPFL, Switzerland
José Luis Marzo, University of Girona, Spain
Ronaldo Menezes, Florida Tech., USA
Roberto Montemanni, IDSIA, Switzerland
Martin Roth, Deutsche Telekom Ltd., Germany
Leon Rothkrantz, Delft University of Technology, The Netherlands
Chien-Chung Shen, University of Delaware, USA
Kwang M. Sim, Hong Kong Baptist University, Hong Kong
Mark C. Sinclair, Royal University of Phnom Penh, Cambodia
George D. Smith, University of East Anglia, UK
Christian Tschudin, University of Basel, Switzerland
Yong Xu, University of Birmingham, UK
Lidia Yamamoto, University of Basel, Switzerland
Franco Zambonelli, University of Modena and Reggio Emilia, Italy

EvoFIN Program Committee

Ernesto Costa, University of Coimbra, Portugal
Carlos Cotta, University of Màlaga, Spain
Ian Dempsey, Pipeline Trading, USA
David Edelman, University College Dublin, Ireland
Philip Hamill, Queen's University Belfast, Ireland
Dietmar Maringer, University of Essex, UK
Robert Schaefer, AGH University of Science and Technology, Poland
Chris Stephens, Universidad Nacional Autonoma de Mexico, Mexico
Ruppa K. Thulasiram, University of Manitoba, Canada

EvoIASP Program Committee

Lucia Ballerini, European Center for Soft Computing, Spain
Bir Bhanu, University of California at Riverside, USA
Leonardo Bocchi, University of Florence, Italy
Alberto Broggi, University of Parma, Italy
Stefano Cagnoni, University of Parma, Italy
Ela Claridge, University of Birmingham, UK
Oscar Cordon, European Center for Soft Computing, Spain
Laura Dipietro, Massachusetts Institute of Technology, USA
Marc Ebner, University of Würzburg, Germany
Daniel Howard, Qinetiq, UK
Mario Koeppen, FhG IPK Berlin, Germany
Evelyne Lutton, INRIA, France

Gustavo Olague, CICESE, Mexico
Riccardo Poli, University of Essex, UK
Stephen Smith, University of York, UK
Giovanni Squillero, Politecnico di Torino, Italy
Kiyoshi Tanaka, Shinshu University, Japan
Ankur M. Teredesai, Rochester Institute of Technology, USA
Andy Tyrrell, University of York, UK
Leonardo Vanneschi, University of Milano-Bicocca, Italy
Robert Vanyi, Siemens PSE, Hungary
Mengjie Zhang, Victoria University of Wellington, New Zealand

EvoInteraction Program Committee

Eric Bonabeau, Icosystem, USA
Praminda Caleb-Solly, University of the West of England, UK
Pierre Collet, Unversite du Littoral, Calais, France
Fang-Cheng Hsu, Aletheia University, Republic of China
Christian Jacob, University of Calgary, USA
Daisuke Katagami, Tokyo Institute of Technology, Japan
Penousal Machado, University of Coimbra, Portugal
Yoichiro Maeda, University of Fukui, Japan
Nicolas Monmarche, Université de Tours, France
Hiroaki Nishino, Oita University, Japan
Ian C. Parmee, University of the West of England, UK
Yago Saez, Universidad Carlos III de Madrid, Spain
Marc Schoenauer, INRIA, France
Daniel Thalmann, EPFL, Switzerland
Leuo-Hong Wang, Aletheia University, Republic of China

EvoMUSART Program Committee

Peter Bentley, University College London, UK
Eleonora Bilotta, University of Calabria, Italy
Tim Blackwell, University of London, UK
Tony Brooks, Aalborg University, Denmark
Paul Brown, University of Sussex, UK
Larry Bull, University of the West of England, UK
Stefano Cagnoni, University of Parma, Italy
Francisco Camara Pereira, University of Coimbra, Portugal
Gianfranco Campolongo, University of Calabria, Italy
Amilcar Cardoso, University of Coimbra, Portugal
John Collomosse, University of Bath, UK
Alan Dorin, Monash University, Australia
Scott Draves, San Francisco, USA
Charlie D. Frowd, University of Stirling, UK
Andrew Gildfind, Royal Melbourne Institute of Technology, Australia
Maria Goga, University of Bucharest, Romania

Nicolae Goga, University of Groningen, The Netherlands
Gary Greenfield, University of Richmond, USA
Carlos Grilo, School of Technology and Management of Leiria, Portugal
Martin Hemberg, Imperial College London, UK
Andrew Horner, University of Science and Technology, Hong Kong
Christian Jacob, University of Calgary, Canada
Janis Jefferies, Goldsmiths College, University of London, UK
Colin Johnson, University of Kent, UK
Francois-Joseph Lapointe, University of Montreal, Canada
William Latham, Art Games Ltd, UK
Matthew Lewis, Ohio State University, USA
Evelyne Lutton, INRIA, France
Bill Manaris, College of Charleston, USA
Ruli Manurung, University of Indonesia, Indonesia
Joao Martins, University of Plymouth, UK
Jon McCormack, Monash University, Australia
James McDermott, University of Limerick, UK
Eduardo R. Miranda, University of Plymouth, UK
Nicolas Monmarché, University of Tours, France
Gary Lee Nelson, Oberlin College, USA
Luigi Pagliarini, Pescara Electronic Artists Meeting, Italy and University of Southern Denmark, Denmark
Pietro Pantano, University of Calabria, Italy
Alejandro Pazos, University of A Coruña, Spain
Rafael Ramirez, Pompeu Fabra University, Spain
Brian J. Ross, Brock University, Canada
Artemis Sanchez Moroni, Renato Archer Research Center, Brazil
Antonino Santos, University of A Coruña, Spain
Jorge Tavares, University of Coimbra, Portugal
Peter Todd, Max Planck Institute for Human Development, Germany
Stephen Todd, IBM, UK
Paulo Urbano, Universidade de Lisboa, Portugal
Jeffrey Ventrella, Independent Artist, USA
Rodney Waschka II, North Carolina State University, USA
Gerhard Widmer, Johannes Kepler University Linz, Austria

EvoSTOC Program Committee

Dirk Arnold, Dalhousie University, Canada
Hans-Georg Beyer, Vorarlberg University of Applied Sciences, Austria
Tim Blackwell, Goldsmiths College, UK
Juergen Branke, University of Karlsruhe, Germany
Ernesto Costa, University of Coimbra, Portugal
Yaochu Jin, Honda Research Institute, Germany
Stephan Meisel, Technical University Braunschweig, Germany
Daniel Merkle, University of Leipzig, Germany

Zbigniew Michalewicz, University of Adelaide, Australia
Martin Middendorf, University of Leipzig, Germany
Ron Morrison, Mitretek Systems, USA
Ferrante Neri, University of Technology of Bari, Italy
Yew Soon Ong, Nanyang Technological University, Singapore
William Rand, Northwestern University, USA
Hendrik Richter, University of Leipzig, Germany
Kumara Sastry, University of Illinois at Urbana Champaign, USA
Ken Sharman, Universidad Politécnica de Valencia, Spain
Anabela Simões, University of Coimbra, Portugal
Christian Schmidt, University of Karlsruhe, Germany

EvoTransLog Program Committee

Christian Bierwirth, University of Halle-Wittenberg, Germany
Karl Doerner, University of Vienna, Austria
Martin J. Geiger, University of Hohenheim, Germany
Jens Gottlieb, SAP, Germany
Jörg Homberger, University of Applied Sciences Kaiserslautern, Germany
Hoong Chuin Lau, Singapore Management University, Singapore
Dirk C. Mattfeld, University of Technology Braunschweig, Germany
Giselher Pankratz, Distance University Hagen, Germany
Christian Prins, Université de Technologie de Troyes, France
Agachai Sumalee, University of Leeds, UK
Theodore Tsekeris, National Technical University of Athens, Greece
Stefan Voss, University of Hamburg, Germany

Sponsoring Institutions

- Universitdad Politécnica de Valencia, Spain
- Instituto Tecnológico de Informática in Valencia, Spain
- Ministerio de Educación y Ciencia, Spain
- The Centre for Emergent Computing at Napier University in Edinburgh, UK

Table of Contents

EvoCOMNET Contributions

Performance of Ant Routing Algorithms When Using TCP 1
Malgorzata Gadomska and Andrzej Pacut

Evolving Buffer Overflow Attacks with Detector Feedback 11
H. Gunes Kayacik, Malcolm Iain Heywood, and A. Nur Zincir-Heywood

Genetic Representations for Evolutionary Minimization of Network Coding Resources 21
Minkyu Kim, Varun Aggarwal, Una-May O'Reilly, Muriel Médard, and Wonsik Kim

Bacterial Foraging Algorithm with Varying Population for Optimal Power Flow 32
M.S. Li, W.J. Tang, W.H. Tang, Q.H. Wu, and J.R. Saunders

An Ant Algorithm for the Steiner Tree Problem in Graphs 42
Luc Luyet, Sacha Varone, and Nicolas Zufferey

Message Authentication Protocol Based on Cellular Automata 52
Angel Martín del Rey

An Adaptive Global-Local Memetic Algorithm to Discover Resources in P2P Networks 61
Ferrante Neri, Niko Kotilainen, and Mikko Vapa

Evolutionary Computation for Quality of Service Internet Routing Optimization 71
Miguel Rocha, Pedro Sousa, Paulo Cortez, and Miguel Rio

BeeSensor: A Bee-Inspired Power Aware Routing Protocol for Wireless Sensor Networks 81
Muhammad Saleem and Muddassar Farooq

Radio Network Design Using Population-Based Incremental Learning and Grid Computing with BOINC 91
Miguel A. Vega-Rodríguez, David Vega-Pérez, Juan A. Gómez-Pulido, and Juan M. Sánchez-Pérez

Evaluation of Different Metaheuristics Solving the RND Problem 101
Miguel A. Vega-Rodríguez, Juan A. Gómez-Pulido, Enrique Alba, David Vega-Pérez, Silvio Priem-Mendes, and Guillermo Molina

A Comparative Investigation on Heuristic Optimization of WCDMA Radio Networks 111
Mehmet E. Aydin, Jun Yang, and Jie Zhang

Design of a User Space Software Suite for Probabilistic Routing in Ad-Hoc Networks 121
Frederick Ducatelle, Martin Roth, and Luca Maria Gambardella

Empirical Validation of a Gossiping Communication Mechanism for Parallel EAs 129
Juan Luís Jiménez Laredo, Pedro Angel Castillo, Ben Paechter, Antonio Miguel Mora, Eva Alfaro-Cid, Anna I. Esparcia-Alcázar, and Juan Julián Merelo

A Transport-Layer Based Simultaneous Access Scheme in Integrated WLAN/UMTS Mobile Networks 137
Hyung-Taig Lim, Seung-Joon Seok, and Chul-Hee Kang

Simplified Transformer Winding Modelling and Parameter Identification Using Particle Swarm Optimiser with Passive Congregation 145
Almas Shintemirov, W.H. Tang, Z. Lu, and Q.H. Wu

A Decentralized Hierarchical Aggregation Scheme Using Fermat Points in Wireless Sensor Networks 153
Jeongho Son, Jinsuk Pak, Hyunsook Kim, and Kijun Han

A Gateway Access-Point Selection Problem and Traffic Balancing in Wireless Mesh Networks 161
Ahmet Cagatay Talay

EvoFIN Contributions

A Genetic Programming Approach for Bankruptcy Prediction Using a Highly Unbalanced Database 169
Eva Alfaro-Cid, Ken Sharman, and Anna I. Esparcia-Alcázar

Multi-objective Optimization Technique Based on Co-evolutionary Interactions in Multi-agent System 179
Rafał Dreżewski and Leszek Siwik

Quantum-Inspired Evolutionary Algorithms for Calibration of the VG Option Pricing Model 189
Kai Fan, Anthony Brabazon, Conall O'Sullivan, and Michael O'Neill

An Evolutionary Computation Approach to Scenario-Based Risk-Return Portfolio Optimization for General Risk Measures 199
Ronald Hochreiter

Building Risk-Optimal Portfolio Using Evolutionary Strategies 208
Piotr Lipinski, Katarzyna Winczura, and Joanna Wojcik

Comparison of Evolutionary Techniques for Value-at-Risk Calculation 218
Gonul Uludag, A. Şima Uyar, Kerem Senel, and Hasan Dag

Using Kalman-Filtered Radial Basis Function Networks to Forecast Changes in the ISEQ Index 228
David Edelman

Business Intelligence for Strategic Marketing: Predictive Modelling of Customer Behaviour Using Fuzzy Logic and Evolutionary Algorithms 233
Andrea G.B. Tettamanzi, Maria Carlesi, Lucia Pannese, and Mauro Santalmasi

EvoIASP Contributions

Particle Swarm Optimization for Object Detection and Segmentation ... 241
Stefano Cagnoni, Monica Mordonini, and Jonathan Sartori

Satellite Image Registration by Distributed Differential Evolution 251
Ivanoe De Falco, Antonio Della Cioppa, Domenico Maisto, Umberto Scafuri, and Ernesto Tarantino

Harmonic Estimation Using a Global Search Optimiser 261
Y.N. Fei, Z. Lu, W.H. Tang, and Q.H. Wu

An Online EHW Pattern Recognition System Applied to Face Image Recognition 271
Kyrre Glette, Jim Torresen, and Moritoshi Yasunaga

Learning and Recognition of Hand-Drawn Shapes Using Generative Genetic Programming 281
Wojciech Jaśkowski, Krzysztof Krawiec, and Bartosz Wieloch

Multiclass Object Recognition Based on Texture Linear Genetic Programming 291
Gustavo Olague, Eva Romero, Leonardo Trujillo, and Bir Bhanu

Evolutionary Brain Computer Interfaces 301
Riccardo Poli, Caterina Cinel, Luca Citi, and Francisco Sepulveda

A Genetic Programming Approach to Feature Selection and Classification of Instantaneous Cognitive States 311
Rafael Ramirez and Montserrat Puiggros

A Memetic Differential Evolution in Filter Design for Defect Detection in Paper Production 320
Ville Tirronen, Ferrante Neri, Tommi Karkkainen, Kirsi Majava, and Tuomo Rossi

Optimal Triangulation in 3D Computer Vision Using a Multi-objective Evolutionary Algorithm 330
Israel Vite-Silva, Nareli Cruz-Cortés, Gregorio Toscano-Pulido, and Luis Gerardo de la Fraga

Genetic Programming for Image Recognition: An LGP Approach 340
Mengjie Zhang and Christopher Graeme Fogelberg

Evolving Texture Features by Genetic Programming 351
Melanie Aurnhammer

Euclidean Distance Fit of Ellipses with a Genetic Algorithm 359
Luis Gerardo de la Fraga, Israel Vite Silva, and Nareli Cruz-Cortés

A Particle Swarm Optimizer Applied to Soft Morphological Filters for Periodic Noise Reduction 367
T.Y. Ji, Z. Lu, and Q.H. Wu

Fast Genetic Scan Matching Using Corresponding Point Measurements in Mobile Robotics 375
Kristijan Lenac, Enzo Mumolo, and Massimiliano Nolich

Overcompressing JPEG Images with Evolution Algorithms 383
Jacques Lévy Véhel, Franklin Mendivil, and Evelyne Lutton

Towards Dynamic Fitness Based Partitioning for IntraVascular UltraSound Image Analysis 391
Rui Li, Jeroen Eggermont, Michael T.M. Emmerich, Ernst G.P. Bovenkamp, Thomas Bäck, Jouke Dijkstra, and Johan H.C. Reiber

Comparison Between Genetic Algorithms and the Baum-Welch Algorithm in Learning HMMs for Human Activity Classification 399
Óscar Pérez, Massimo Piccardi, Jesús García, Miguel Ángel Patricio, and José Manuel Molina

Unsupervised Evolutionary Segmentation Algorithm Based on Texture Analysis 407
Cynthia Beatriz Pérez and Gustavo Olague

Evolutionary Approaches for Automatic 3D Modeling of Skulls in Forensic Identification 415
Jose Santamaría, Oscar Cordón, and Sergio Damas

Scale Invariance for Evolved Interest Operators 423
Leonardo Trujillo and Gustavo Olague

Application of the Univariate Marginal Distribution Algorithm to Mixed Analogue - Digital Circuit Design and Optimisation 431
Lyudmila Zinchenko, Matthias Radecker, and Fabio Bisogno

EvoINTERACTION Contributions

Interactive Texture Design Using IEC Framework 439
Tsuneo Kagawa, Yukihide Tamotsu, Hiroaki Nishino, and Kouichi Utsumiya

Towards an Interactive, Generative Design System: Integrating a 'Build and Evolve' Approach with Machine Learning for Complex Freeform Design 449
Azahar T. Machwe and Ian C. Parmee

An Interactive Graphics Rendering Optimizer Based on Immune Algorithm 459
Hiroaki Nishino, Takuya Sueyoshi, Tsuneo Kagawa, and Kouichi Utsumiya

Human Mosaic Creation Through Agents and Interactive Genetic Algorithms Applied to Videogames Movements 470
Oscar Sanjuán, Gloria García, Yago Sáez, and Cristobal Luque

EvoMUSART Contributions

Self-organizing Bio-inspired Sound Transformation 477
Marcelo Caetano, Jônatas Manzolli, and Fernando Von Zuben

An Evolutionary Approach to Computer-Aided Orchestration 488
Grégoire Carpentier, Damien Tardieu, Gérard Assayag, Xavier Rodet, and Emmanuel Saint-James

Evolution of Animated Photomosaics 498
Vic Ciesielski, Marsha Berry, Karen Trist, and Daryl D'Souza

Environments for Sonic Ecologies 508
Tom Davis and Pedro Rebelo

Creating Soundscapes Using Evolutionary Spatial Control 517
José Fornari, Adolfo Maia Jr., and Jônatas Manzolli

Toward Greater Artistic Control for Interactive Evolution of Images and Animation 527
David A. Hart

Evolutionary Assistance in Alliteration and Allelic Drivel 537
Raquel Hervás, Jason Robinson, and Pablo Gervás

Evolutionary GUIs for Sound Synthesis 547
James McDermott, Niall J.L. Griffith, and Michael O'Neill

Evolving Music Generation with SOM-Fitness Genetic Programming ... 557
Somnuk Phon-Amnuaisuk, Edwin Hui Hean Law, and Ho Chin Kuan

An Automated Music Improviser Using a Genetic Algorithm Driven Synthesis Engine 567
Matthew John Yee-King

Interactive GP with Tree Representation of Classical Music Pieces 577
Daichi Ando, Palle Dahlsted, Mats G. Nordahl, and Hitoshi Iba

Evolutionary Methods for Melodic Sequences Generation from Non-linear Dynamic Systems 585
Eleonora Bilotta, Pietro Pantano, Enrico Cupellini, and Costantino Rizzuti

Music Composition Using Harmony Search Algorithm 593
Zong Woo Geem and Jeong-Yoon Choi

Curve, Draft, and Style: Three Steps to the Image 601
Olgierd Unold and Maciej Troc

GISMO2: An Application for Agent-Based Composition 609
Yuta Uozumi

EvoSTOC Contributions

Variable-Size Memory Evolutionary Algorithm to Deal with Dynamic Environments 617
Anabela Simões and Ernesto Costa

Genetic Algorithms with Elitism-Based Immigrants for Changing Optimization Problems 627
Shengxiang Yang

Triggered Memory-Based Swarm Optimization in Dynamic Environments 637
Hongfeng Wang, Dingwei Wang, and Shengxiang Yang

Experimental Comparison of Replacement Strategies in Steady State Genetic Algorithms for the Dynamic MKP 647
A. Şima Uyar

Understanding the Semantics of the Genetic Algorithm in Dynamic Environments 657
Abir Alharbi, William Rand, and Rick Riolo

EvoTRANSLOG Contributions

Simultaneous Origin-Destination Matrix Estimation in Dynamic Traffic Networks with Evolutionary Computing 668
Theodore Tsekeris, Loukas Dimitriou, and Antony Stathopoulos

Evolutionary Combinatorial Programming for Discrete Road Network Design with Reliability Requirements 678
Loukas Dimitriou, Theodore Tsekeris, and Antony Stathopoulos

Intelligent Traffic Control Decision Support System 688
Khaled Almejalli, Keshav Dahal, and M. Alamgir Hossain

An Ant-Based Heuristic for the Railway Traveling Salesman Problem 702
Petrica C. Pop, Camelia M. Pintea, and Corina Pop Sitar

Enhancing a MOACO for Solving the Bi-criteria Pathfinding Problem for a Military Unit in a Realistic Battlefield 712
Antonio Miguel Mora, Juan Julian Merelo, Cristian Millan, Juan Torrecillas, Juan Luís Jiménez Laredo, and Pedro A. Castillo

GRASP with Path Relinking for the Capacitated Arc Routing Problem with Time Windows 722
Mohamed Reghioui, Christian Prins, and Nacima Labadi

Multi-objective Supply Chain Optimization: An Industrial Case Study 732
Lionel Amodeo, Haoxun Chen, and Aboubacar El Hadji

Scheduling a Fuzzy Flowshop Problem with Flexible Due Dates Using Ant Colony Optimization 742
Sezgin Kilic

Author Index 753

Performance of Ant Routing Algorithms When Using TCP

Malgorzata Gadomska and Andrzej Pacut

Institute of Control and Computation Engineering
Warsaw University of Technology
00-665 Warsaw, Poland

Abstract. It is commonly believed that Ant Routing algorithms cannot be applied with the TCP transport layer. We show that, contrary to this belief, TCP in the transport layer still enables the adaptive algorithms to extend the range of load levels under which they can find efficient routing policies.

1 Introduction

With the continuing growth of the Internet and other communication networks, an efficient traffic organization is becoming increasingly important. In particular, there is a strong need for routing algorithms that can distribute data traffic across multiple paths and quickly adapt to changing conditions such as changes in the network load distribution, or changes in the network topology. In particular, packet routing algorithms play a critical role in the network performance, especially in terms of throughput and transmission delay.

In recent years, many such adaptive routing algorithms have been proposed. We concentrate on the *ant routing* algorithms based on swarm intelligence. This approach has already proved its efficiency in various simulated environments. Ant algorithms do not require supervision, and their distributed form makes them well applicable to the routing problem. Ant routing algorithms are typically considered with UDP in the transport layer. To be useful in a real Internet environment, such routing algorithm should perform well also with the TCP, since most applications use the TCP as a transport layer protocol to ensure a reliable transmission.

Adaptive algorithms may be characterized as multipath routing algorithms, since packets belonging to the same TCP session can be sent along different routes. The path selection can be done, for instance, according to a stochastic policy, where the probability of choosing a particular path is proportional to a quality of this path. Under such conditions, the packets sent earlier can reach their destination after the packets sent later, but along a "better" path. Consequently, packet reordering is not a sign of abnormality, yet it is interpreted by the TCP in such way. Most standard implementations of TCP perform poorly when packets are reordered as a consequence of varying packet routes.

The aim of this paper is to investigate the performance of ant routing algorithms with the TCP in the transport layer. We compare this performance to the one obtained with the UDP. We also investigate an influence of various

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 1–10, 2007.
© Springer-Verlag Berlin Heidelberg 2007

implementations of the TCP on the adaptive routing algorithms, including modifications that make it more robust to packet reordering. We show that also when using the TCP in the transport layer, the adaptive algorithms in most cases extend the range of load levels under which the algorithms succeed in finding the efficient routing policies, as compared to the Shortest Path algorithm.

The paper is organized as follows. In Sec. II, we shortly introduce the swarm intelligence approach to adaptive routing and describe the ant-routing algorithms. In Sec. III, we briefly describe the TCP and some of its modifications. The simulation environment is described in Sec. IV. Performance of ant algorithms when using the TCP and UDP is compared in Sec. V, followed by an analysis of data distribution influence in Sec. VI, and the TCP version influence in Sec. VII. Section VIII outlines the performance of ant algorithms under a packet loss. Section IX concludes the paper.

2 Ant Routing Algorithms

Ant algorithms were inspired by observations of real ant colonies and were first proposed by Dorigo, Maniezzo, and Colorni in 1991 [5] as a multi-agent approach to the traveling salesman problem (TSP) and the quadratic assignment problem (QAP). The ants are capable of finding the shortest path from the nest to food sources. While traveling, they deposit a chemical substance (a pheromone), and its high concentration guides the other ants to form the optimal paths.

The first ant routing algorithm was proposed by Schoonderwoerd *et al.* in 1996 [8]. Their ABC algorithm could be applied only to symmetric circuit-switched networks. The AntNet algorithm introduced by Dorigo and di Caro in 1998 [6] has no such limitations and is the first ant-routing algorithm investigated here. Various modifications of AntNet have been developed in the following years, e.g. a limitation of the number of ants in the network [10], or a non-greedy policy determination. The Adaptive Swarm-based Routing (ASR) proposed in 2004 for packet-switched networks by Yong, Guang-Zhou, and Fan-Jun [11] is the second ant routing algorithm used in our studies.

Both in AntNet [6] and in ASR [11] there are two types of simple agents (ants): the *forward ants* that explore the network in order to find paths, and the *backward ants* that use information collected by the forward ants to improve the routing policies. Every network node is assigned a *routing table* and a statistical *traffic model* including some local traffic statistics. The routing table T_k stores the probabilities $t_k(d, n)$ for each neighbor node n and each destination node d, used to determine the probabilistic routing policy. Both the routing tables and the traffic models are calculated iteratively during the normal operation of the network. In AntNet, the traffic model at each node k consists of the estimates of the expected value and the variance of the trip time from k to the destination node d. In ASR, for every destination node d and neighbor node n, the last two estimates of the trip time from k via n to the destination node d are stored. The second significant difference between these algorithms is the way of updating the routing probabilities stored in the routing table. Details of the algorithms and their performance comparison can be found in [7].

3 The TCP and Its Modifications

When using the TCP in the transport layer, the network dynamics is mainly determined by TCP's error and congestion control mechanisms ([9]), which employ the *slow start* and *congestion avoidance* algorithms, or the duplicate acknowledgment (DUPACK) mechanism and the fast retransmit algorithm. These mechanisms may slow down the ant algorithms to such degree that it may become useless for the routing problem. This issue is being investigated here.

The design of TCP's error and congestion control mechanisms assume that a packet loss is an indication of a network congestion. Therefore, when a loss is detected, the TCP sender reduces its transmission rate by decreasing its *congestion window*, which constraints the number of packets that can be sent. TCP uses two strategies for detecting the packet loss. The first one is based on the senders retransmission time-out (RTO) expiration. The second mechanism originates at the receiver, which observes the sequence numbers of packets it receives and generates a DUPACK for every out-of-order packet.

For multipath routing algorithms, packet reordering is not necessarily an indication of congestion, since it may occur in their normal operation. Several mechanisms that address TCP's lack of robustness to packet reordering have been recently proposed ([2], [3], [12]). For instance in the TCP-PR protocol ([3]) it is proposed to neglect DUPACKs entirely and to rely solely on timers to detect drops: if the ACK for a packet has not arrived and the time elapsed since the packet is sent exceeds a retransmission threshold then the packet is assumed to be lost. The threshold should adapt to the changing network conditions in such way that a packet is retransmitted only if it has been really lost. This TCP modification has been investigated in our paper.

4 Simulation Environment

Our experiments were performed using a custom-made event-driven network simulator ADNET implemented in Java. This simulator enables dynamic (during the simulation) definitions of the packet generation scenarios, contrary to other popular simulators like NS-2. ADNET implements the main network layers: transport, network and application layer. We implemented AntNet, ASR and the Shortest Path algorithm in the network layer. The transport layer is equipped with UDP and a few versions (TCP Tahoe, TCP Reno, TCP-PR) of the TCP. Every router in the network generates traffic in portions of data according to independent Poisson processes. Each portion consist of packets, whose number is either constant or is distributed according to the geometric distribution. The destination nodes of the packets are chosen randomly.

The methodology presented in this paper was applied to the grid network structure proposed by Boyan and Littman [4] (Fig. 1, upper left), to the NASK (Scientific and Academic Network, Poland) network (Fig. 1, upper right) and the NTT (Nippon Telephone Telegraph, Japan) network used in [1] (Fig 1, bottom).

We used two closely related quality indicators of the learned policy, namely, the average packet delay, and the average number of packets in the network.

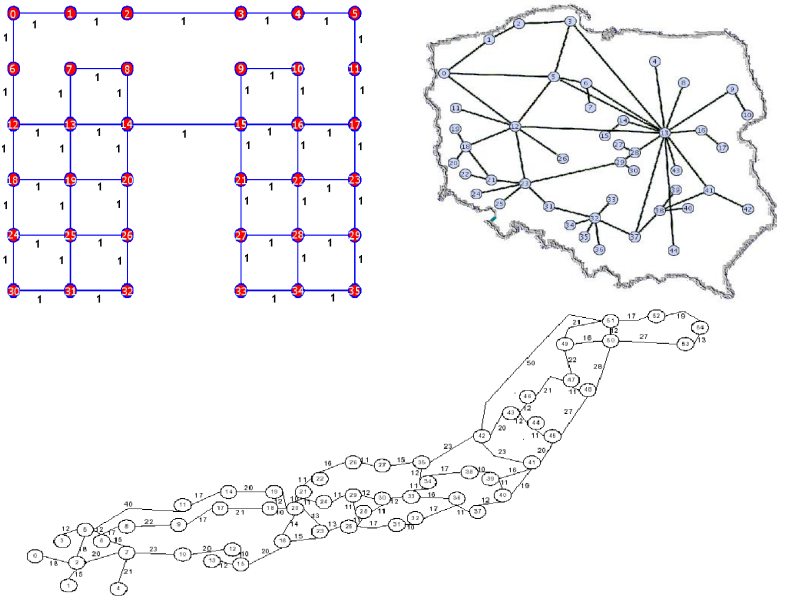


Fig. 1. Network structures used in simulations: B-L network [4] (upper left), NASK network (upper right) and NTT network (lower figure)

By the learning time we understand the time necessary for the average packet delay to stabilize. To express the network load we used relative units defined as the maximum load level under which Dijkstra's Shortest Path (SP) algorithm still works for a given network structure and a given load pattern. For short, we called such the relative unit *the Dijkstra* (D). In other words, the network load is 1.5 D if it is 50% higher than the maximum load under which the given network still works with the SP policy. Therefore, all loads above 1D cannot be served by the SP policy. Certainly, the actual load of 1D depends on the network structure (geometry, capacity, queues structure) and the load pattern time-space distribution (constant vs random in time, uniform vs. non-uniform in space).

The network load level is understood as the load level generated by the data packets. This means that the nominal load level presented in the figures does not take into account the additional load generated by ants. This additional load is yet considered to determine the actual network traffic. All the described experiments where performed under a constant load.

5 TCP vs. UDP - Ants Performance Comparison

Our experiments did not show significant differences in the adaptation process of ant algorithms for different transport layer protocols under *low load levels* (below 1D) (Fig. 2). While the learning time is longer when using the TCP, the

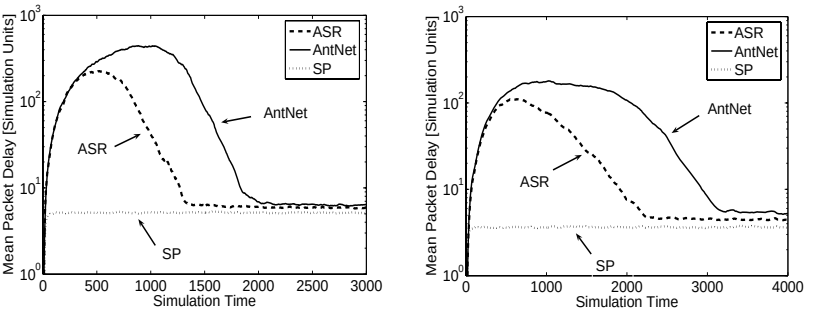


Fig. 2. Low load level (0.67 D) performance of SP, ASR and AntNet when using UDP (left) and TCP (right) for B-L network. This and all the following graphs show the averages over 10 simulations. Note a longer learning time for TCP.

maximal average packet delay during learning is slightly lower. Independently of the transport layer protocol, both ant routing algorithms converged quite fast to the policies similar to the Shortest Path. The ASR algorithm lead to shorter learning times than AntNet (see also [7]).

Advantages of adaptive algorithms can be seen under *high load levels* (higher than 1D), when the SP policy does not work efficiently. In most cases, adaptive algorithms used with the UDP extend the range of load levels under which the algorithms succeed in finding the efficient routing policies ([7]). The situation complicates for TCP due to error and congestion control mechanisms which may cause the adaptation to be less efficient. Experiments with the AntNet applied to the NTT network show that the *convergence range*, namely the range of load levels under which the algorithm manages to learn the proper routing policies, decreases when using TCP (Fig. 3). It can be seen that TCP may cause AntNet to diverge under a load for which UDP still enables to find the efficient routing policies (Figs. 4 and 5 for B-L and NASK networks). For the UDP, the learning

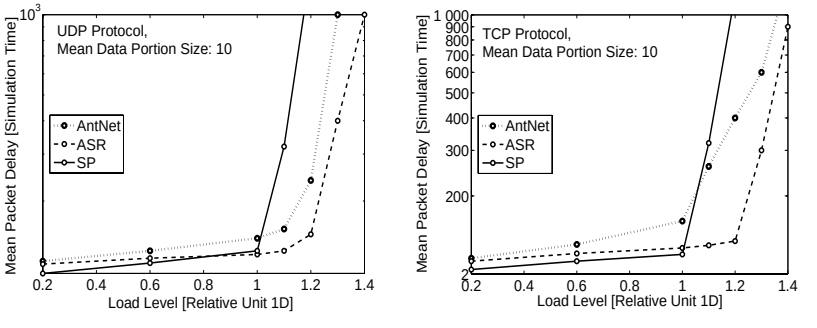


Fig. 3. Convergence ranges of SP, ASR and AntNet when using UDP (left) and TCP (right) for NTT network. Note different packet delay time ranges.

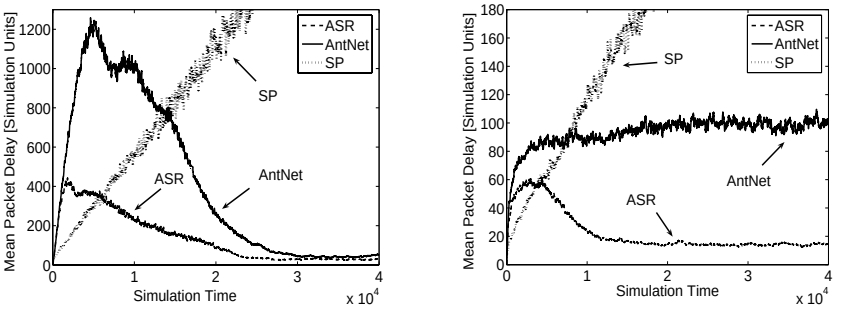


Fig. 4. High load level (1.11D) performance of SP, ASR and AntNet when using UDP (left) and TCP (right) for B-L network. Note that SP diverges for both protocols, and AntNet diverges for TCP.

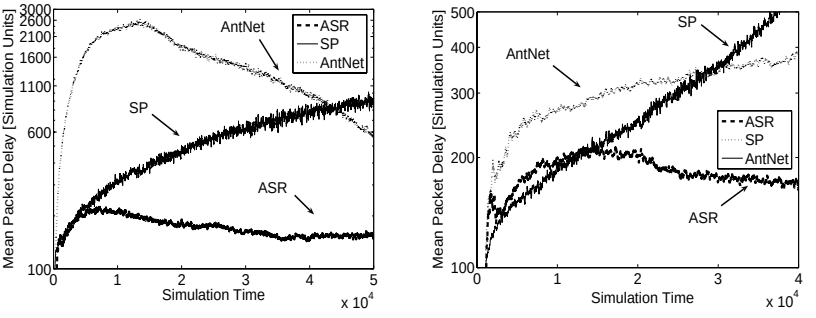


Fig. 5. High load level (1.2 D) performance of SP, ASR and AntNet when using UDP (left) and TCP (right) for the NTT network. As for B-L network, SP diverges for both UDP and TCP, and AntNet diverges for TCP.

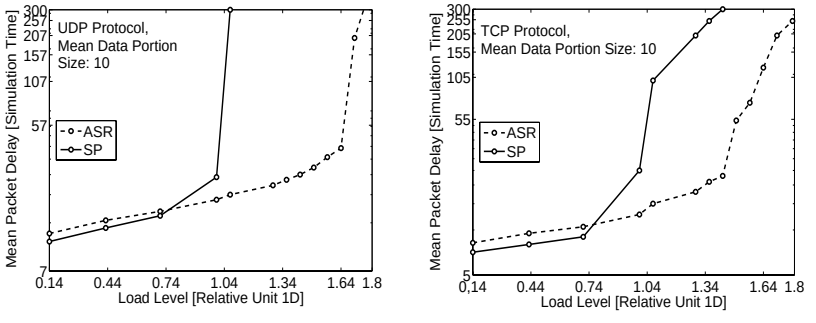


Fig. 6. Comparison of the convergence ranges of SP and ASR when using UDP (left) and TCP (right) for NASK network

times are longer and the maximum packet delays during learning are higher, which is especially visible for the B-L network (Fig. 4). On the other hand, the transport layer protocol does not significantly influence the performance of the

ASR algorithm. The range of load levels that provide efficient routing does not change markedly for any of the tested networks (Figs. 3 and 6).

Concluding, the ASR algorithm performs very well for both the UDP and the TCP. In AntNet, the routing probabilities are not directly dependent of the packet delays, so the higher is the load level, the more frequently packet reordering occurs. As a result, many spurious retransmissions can be observed in the network, even after the learning process is complete. On the other hand, the routing probabilities in ASR are inversely proportional to the packet delays. Therefore, the reordering rate decreases during the learning time, as the travels along routes with similar routing probabilities should result in similar delays. Since the ASR algorithm performs better than the AntNet (see also [7]), we illustrate further results using only the ASR experiments.

6 Influence of the Data Portion Distribution

The influence of the data portion distribution parameters on the performance of ant algorithms was also tested. It can be seen both for low and high load levels, that the bigger are the portions of data, the lower is the maximum packet delay during the learning process (Fig. 7) . However, increase of the packet portion sizes results in the increase of the mean packet delay, when the network stabilizes. This is due to the fact that for large data portions, many packets appear at a router at once. On the other hand, small data portions cause the load level to be more uniformly distributed over time.

It is worth noticing that under high load levels, an increase of the data portion size makes the learning time increase. On the other hand, under low load levels, large data portions result in longer periods between the successive data generations. As an effect, periodical growths and drops of the load level can be observed in the network. Moreover, also the mean packet delays fluctuate in time. This effect is especially well seen for B-L network, when the packets are generated in portions of 1000 (Fig. 7 - left).

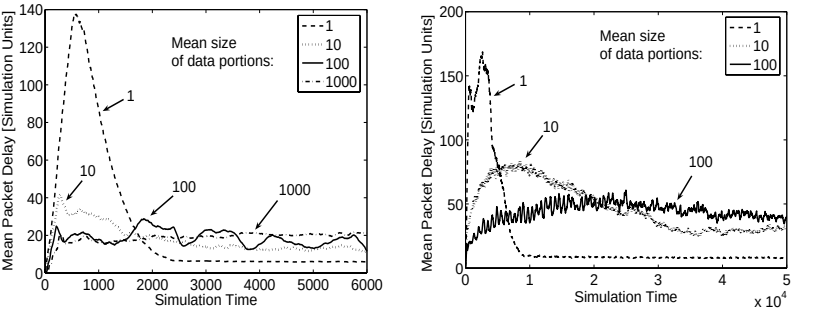


Fig. 7. Influence of the data portion's size on ASR performance when using TCP, for low (0.67 D) load level (left) and high (1.11 D) load level (right) for B-L network

7 Influence of the TCP Version

We tested the differences between the influence of TCP Tahoe, TCP Reno and TCP-PR ([3]) on the ant-routing performance. During the experiment, packets are not being lost, hence all duplicate acknowledgments are unnecessary and are the result of packet reordering.

When using TCP Reno under high load levels, the ASR algorithm does not manage to find efficient routing policies (Fig. 8). For a load level of 1.3 D the mean packet delay for the B-L network increases during the simulation and does not settle. Also, TCP Tahoe made the ASR algorithm diverge in some simulation runs, making 1.3 D the upper limit load for the ASR to converge when using TCP Tahoe for the B-L network. Under the same load of 1.3 D, the TCP-PR ensured the convergence of the learning process (Fig. 8). Neglecting DUPACKS results in decreasing the number of spurious retransmissions caused by packet reordering. As an effect, the load level is reduced and the ASR algorithm manages to find efficient routing policies. The packet reordering rate varies over time being the

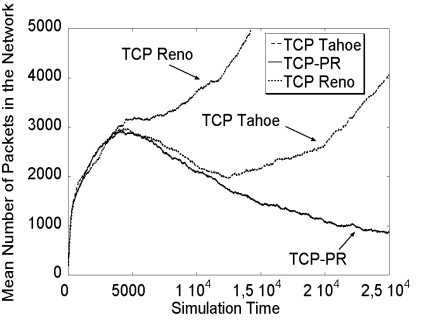


Fig. 8. Influence of the TCP version on ASR algorithm under load level of 1.3 D for the B-L network. For TCP Tahoe some of simulation the runs diverge, thus the averaged graph indicates first a converging tendency, then diverges.

highest during the adaptation process, since the exploration is the highest at that time and the efficient routes are unknown. When the load is constant in time, after the learning process is finished, the reordering rate decreases because the ASR quality is measured by the delivery time. Therefore, the packets are sent mainly along the shortest paths.

8 Performance Under Packet Loss

The TCP, unlike the UDP, is a connection oriented protocol that guarantees reliable data transmission. Here we test a performance of ant routing under

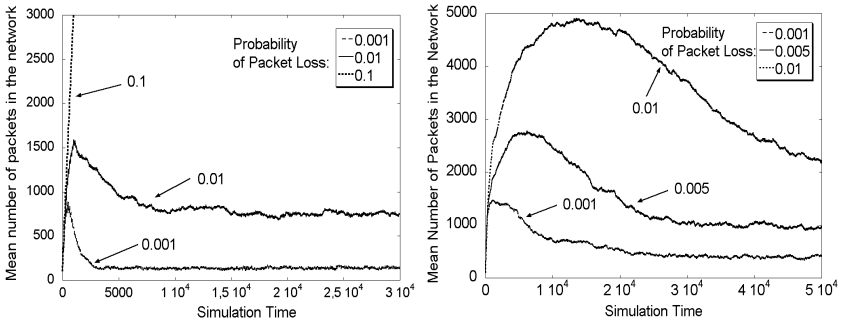


Fig. 9. ASR performance with packet loss under TCP Tahoe, for low (0.67 D) load level (left) and high (1.11 D) load level (right) for B-L network

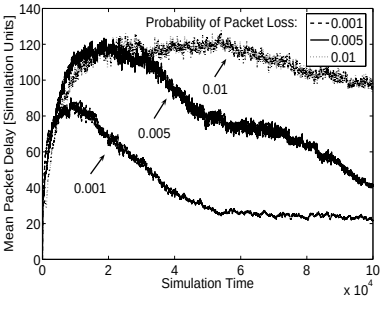


Fig. 10. ASR performance with packet loss, using TCP Tahoe, for high (1.43 D) load level for NASK network

packet loss for TCP. We assumed that each router, independently of other, can loose a packet with a certain probability.

Both under low and under high load levels an increase of the probability of loosing a packet makes the maximum of the average number of packets in the network during the learning process to increase (Fig. 9). Moreover, the learning process lasts longer, since the probability of loosing a packet concerns all packets, including the ant packets. As a result, less ants manage to reach their destination and the process of propagating the information about efficient routes slows down. The higher is the probability of loosing a packet, the higher is the mean packet delay after the learning process (Fig. 10). It is a result of more frequent retransmissions of the lost packets. Since every lost packet must be retransmitted, the packet delivery time from the moment the packet was first sent until the packet successfully reaches its destination, becomes longer. It can be seen that under low load level for the B-L network, the loss probability equal to 0.1 makes the ASR diverge. Most of the packets are delayed by the TCP agents and the repeated retransmissions cause the network congestion (Fig. 9).

9 Conclusions and Further Work

We performed our simulations for the benchmark B-L network, as well as for certain actual structures like the Polish Scientific and Academic Network NASK or the skeleton network of Japan NTT. We showed that the use of TCP is not prohibitive to parallel routing, including the ant algorithms. While the TCP sets higher demands on the adaptation processes, it is still possible to extend the load range of the network. The TCP can even lower the adaptation time.

We currently develop modifications of ant algorithms directed into using the properties of the existing TCPs, and modifications of the TCP that use the information collected by ants to improve their routing performance.

References

1. B. Barn, R. Sosa, "AntNet: Routing Algorithm for Data Networks based on Mobile Agents", http://aepia.dsic.upv.es/revista/numeros/12/baran.pdf
2. E. Blanton, M. Allman, "On making TCP more robust to packet reordering", *ACM Computer Communications Review*, 32, 2002.
3. S. Bohacek, J. P. Hespanha, J. Lee, C. Lim, K. Obraczka, "TCP-PR: TCP for Persistent Packet Reordering", Tech. Rep., Univ. of California Santa Barbara, 2003
4. J. A. Boyan and M. L. Littman, "Packet Routing in Dynamically Changing Networks: A Reinforcement Learning Approach", *Advances in Neural Information Processing Systems*, volume 6, pages 671678, Morgan Kaufmann Publishers, Inc., 1994.
5. M. Dorigo, V. Maniezzo, and A. Colorni, "Positive feedback as a search strategy", *Technical Report 91-016*, Politecnico di Milano, Dipartimento di Elettronica, 1991.
6. M. Dorigo and G. di Caro, "AntNet: Distributed Stigmergetic Control for Communications Networks", *Journal of Artirtificial Intelligence Research 9*, 317-365, 1998.
7. A. Pacut, M. Gadomska, A. Igielski, "Ant-Routing vs. Q-Routing in Telecommunication Networks", *Proceedings of the 20-th ECMS Conference*, 2006.
8. R. Schoonderwoerd, O. Holland, J. Bruten, and L. Rothkrantz, "Ant-based load balancing in telecommunications networks", *Adaptive Behavior, 5(2)*, 169-207, 1996.
9. W. R. Stevens, "TCP/IP Illustrated, Volume I, II", Addison-Wesley, 1994.
10. F. Tekiner, F. Ghassemlooya, S. Al-khayattb, "The Antnet Routing Algorithm - A Modified Version", *Argentine Symposium on Artificial Intelligence*
11. Lu Yong, Zhao Guang-zhou, Su Fan-jun, "Adaptive swarm-based routing in communication networks", *Journal of Zhejiang Univ. SCIENCE, 5(7)*, 867-872, 2004.
12. N. Zhang, B. Karp, S. Floyd, L. Peterson, "RR-TCP: A reordering-robust TCP with DSACK", *Tech. Rep. TR-02-006*, ICSI, Berkeley, CA

Evolving Buffer Overflow Attacks with Detector Feedback

H. Gunes Kayacik, Malcolm I. Heywood, and A. Nur Zincir-Heywood

Faculty of Computer Science, Dalhousie University
6050 University Avenue. Halifax. NS. Canada
{kayacik,mheywood,zincir}@cs.dal.ca

Abstract. A mimicry attack is an exploit in which basic behavioral objectives of a minimalist 'core' attack are used to design multiple attacks achieving the same objective from the same application. Research in mimicry attacks is valuable in determining and eliminating detector weaknesses. In this work, we provide a process for evolving all components of a mimicry attack relative to the Stide (anomaly) detector under a Traceroute exploit. To do so, feedback from the detector is directly incorporated into the fitness function, thus guiding evolution towards potential blind spots in the detector. Results indicate that we are able to evolve mimicry attacks that reduce the detector anomaly rate from ~67% of the original core exploit, to less than 3%, effectively making the attack indistinguishable from normal behaviors.

1 Introduction

Our objective is to develop an automated process for building "white-hat" attackers within a mimicry context [1,2,3,4]. By 'mimicry' we assume the availability of the 'core' attack, where this establishes a series of behavioral objectives associated with the exploit [5,6]. The goal of the automated white hat attacker will therefore be to establish as many specific attacks corresponding to the exploit associated with the 'core' attack as possible. Candidate mimicry attacks will take the form of system call sequences that can avoid detection or at least minimize the anomaly rate at the corresponding detector. By "white hat", we imply that the underlying objective is to use the attacks to improve the design of corresponding detectors.

Previous research has established the suitability of evolutionary computation as an appropriate process for automating the parameterization of buffer overflow attacks [5], and for designing a generic buffer overflow attack itself [6]. In this work, we extend the approach to explicitly incorporate feedback from the anomaly detector. Moreover, previous instances of evolved attacks were designed within the context of a virtual vulnerability and verified against the Snort signature based detector post training [5,6]. Conversely, this work provides attacks that are specific to a real application vulnerability under the more advanced behavioral anomaly detection paradigm. To do so, evolution is guided towards attacks that are able to make use of unforeseen weaknesses in the detector, thus

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 11–20, 2007.
© Springer-Verlag Berlin Heidelberg 2007

providing the basis for improvements in detector design or vulnerability testing. Relative to earlier works on mimicry attack generation (against behavioral anomaly detectors) [1,2,3,4], the adoption of an evolutionary approach to designing attacks demonstrates that it is no longer necessary to rely on privileged detector information to establish valid attacks. As such, we believe that the ensuing attacks are more reflective of vulnerabilities that are likely to be developed by a "would be" attacker in practice.

In the following we detail the process used to configure the anomaly detector, and characterize the vulnerable application, Section 2, before introducing the evolutionary mimicry attack framework, Section 3. Results are presented in Section 4, in which attacks are successfully designed with anomaly rates less than three percent; in effect making them indistinguishable from normal behavior. Moreover, specific recommendations are made regarding the construction of appropriate search operators. Conclusions are drawn in Section 5, with the case made for the coevolution of white hat attacks and detectors.

2 Detector and Vulnerable Application

The general goal of this work is to demonstrate that real white hat exploits may be developed under an evolutionary computation paradigm given a mimicry attack model. To be of relevance to vulnerability testing of a specific detector, behavioral goals of an exploit are augmented with feedback from the detector itself. Thus, for vulnerabilities to exist in the detector we aim to evolve programs that provide the desired exploit whilst minimizing the detector anomaly rate. Unlike previous work on mimicry attack generation, no inside knowledge is utilized in identifying weaknesses in the detector [1,2,3,4]. With these general guidelines in mind, we first provide the background to the detector on which vulnerability testing will be conducted, introduce the configuration process, and establish how a successful attack will be recognized.

2.1 Anomaly Based Intrusion Detection

Anomaly based intrusion detection systems (IDS) attempt to build models of normal user behavior and use this as the basis for detecting suspicious activities. This way, known and unknown (i.e. new) attacks can be detected as long as the attack behavior deviates sufficiently from the normal behavior. Needless to say, if the attack is sufficiently similar to the normal behavior, it may not be detected. However, user behavior itself is not constant, thus even the normal activities of a user may raise alarms. In this work, Stide was used as the target anomaly detector [7]; where a wide range of related research performed in vulnerability or penetration testing has employed Stide on the basis of it's open source availability and behavioral approach to anomaly detection [1,2,3,4]. That is to say, rather than taking the 'signature' based approach to detection[1], the behavioral

[1] Where evading signature based detectors using mimicry methods is already considered straightforward [1].

methodology develops a model for normal behavior for specific services using *a priori* supplied system call(s). As such the ensuing detector does not require a direct match between modeled behavior and an attack for recognition to take place as per a signature based detector, but returns a percent anomaly rate. The user (typically a system administrator) is then free to interpret the anomaly rate as representing an attack or not, usually by specifying a threshold. It is this anomaly rate that will be used to guide the evolutionary process towards any weaknesses in the detector. Section 2.2 details how Stide was configured within the context of the vulnerable application.

2.2 Vulnerable Application and Configuration of Stide

In the following, Traceroute is employed as the vulnerable application. Traceroute is used to determine the routing path between a source and destination by sending a set of control packets to the destination with increasing time-to-live values. A typical use of traceroute involves providing the destination IP, whereas the application returns information on the route taken between source and destination.

Redhat 6.2 was shipped with Traceroute version 1.4a5, where this is susceptible to a local buffer overflow exploit that provides a local user with super-user access [8]; hereafter the 'core' attack. The attack exploits a vulnerability in malloc chunk, and then uses a debugger to determine the correct return address to take control of the program. As indicated above, an anomaly detector is used in conjunction with a threshold (anomaly rate) such that detection and false positive rates are optimized. The objective of the attacker is to build attacks that return anomaly rates below the threshold characterizing normal behavior. One approach to establishing a safe detector threshold might be to set the threshold to zero. However, this would result in far too many false positive alarms. That is to say, in practice, the normal behavior model of the detector cannot cover all possible user scenarios, as configuration is only conducted over a subset of behavioral traces. However, we also note that the available configurations of the Traceroute application are also limited; thus only a small number of use cases are sufficient to provide a complete set of system call sequences to characterize normal behavior[2]. In this work we consider two scenarios. Scenario 1 configures Stide using a single use case, the 'nist' domain, as per previous work in mimicry research [1,2,3,4]. The principle motivation being that if attacks can be designed against a minimalist Stide configuration, then designing attacks for a typical configuration will be easier. Scenario 2 configures Stide using 5 use cases, as follows: search engines; local servers; a non existent host; the local host; and the application help screen. The motivation in this case being that an attacker would not have access to the database of normal behaviors that Stide uses to characterize normal behavior, thus being able to evade detection under typical operating conditions is more reflective of an attackers' objectives in practice.

[2] Stide builds a behavioral model based on system call sequences alone; no use is made of arguments, thus avoiding any sensitivity to specific system call parameters [7].

Moreover, previous research has concentrated exclusively on scenario 1, whilst always assuming access to internal data structures of the detector. Conversely in this work we are interested in compromising the detector without recourse to privileged detector information, thus more reflective of the mode of operation associated with a hacker in practice.

3 Evolutionary Framework for Mimicry Attack Generation

The case for using evolutionary computation in a mimicry attack context has previously been made with respect to: the utility of code bloat for obfuscation of malicious code; freedom in defining fitness functions most appropriate to the application domain; and solutions taking the direct form of the attack itself [6]. Such a combination of properties precludes the utilization of other machine learning paradigms, such as neural networks, decision trees, and kernel methods. However, the feasibility of evolving attacks was previously established in terms of a hypothetical application, and did not incorporate the detector in any way. As a consequence, vulnerability testing is only appropriate under a signature based detection paradigm.

Behavioral based anomaly detectors are configured with respect to specific applications. Thus, vulnerability testing is also carried out with respect to a specific application. The first step of our framework for evolving attacks against such behavioral detectors is to identify an instruction set that is not likely to be immediately recognized as anomalous by the detector. Secondly, a fitness function needs crafting that focus on the relevant behavioral properties of the 'core' exploit (Traceroute in this case). Finally, we need to define the mechanism for integrating both the behavioral components and the detector feedback into an overall fitness function. Subsections 3.1 and 3.2 detail the framework used to address these points, with Subsection 3.3 summarizing the evolutionary model employed in this work.

3.1 Identifying Instruction Set

In order to minimize the likelihood of the exploit being detected, we restrict the instruction set from which attacks are evolved to those appearing in the target application (Traceroute). Table 1 details the frequency of the top twenty system calls executed by the Traceroute application. This accounts for over 90% of the normal instruction set. The system calls used to construct attacks will therefore consist of the top 15 from this list. Note that such an approach does not require any knowledge of the detector. Establishing the system calls associated with the application merely implies that a diagnostic tool[3] is deployed to identify those sent to the operating system by the vulnerable application during execution.

[3] In this case the Strace diagnostic tool was employed, http://sourceforge.net/projects/strace/

Table 1. Frequency of top 20 system calls

System Call	Occurence	Frequency	System Call	Occurence	Frequency
gettimeofday	220	16.73%	mprotect	34	2.59%
write	142	10.8%	socket	29	2.21%
mmap	113	8.59%	recvfrom	28	2.13%
select	99	7.53%	brk	27	2.05%
sendto	99	7.53%	fcntl	26	1.98%
close	93	7.07%	connect	20	1.52%
open	86	6.54%	ioctl	15	1.14%
read	75	5.7%	uname	14	1.06%
fstat	73	5.55%	getpid	12	0.91%
munmap	49	3.73%	time	10	0.76%

3.2 Fitness Function

The original attack contains a standard shellcode, which uses the execve system call to spawn a UNIX shell upon successful execution. Since traceroute never uses an execve system call (Table 1), the original attack can be easily detected. To this end, we employ a different attack strategy by eliminating the need to spawn a UNIX shell. Most programs typically perform I/O operations, in particular open, write to / read from and close files. Table 1 demonstrates that traceroute frequently uses open / write / close system calls. We therefore recognize that performing the following three steps establish the goals of the original shell code attack:

1. Open the UNIX password file ("/etc/passwd");
2. Write a line, which provides the attacker a super-user account that can login without a password;
3. Close the file.

The objective of the evolutionary search process is to discover a sequence of system calls that perform the above three steps in the correct order (i.e. the attack cannot write to a file that it has not opened) while minimizing the anomaly rate from Stide. Hence the fitness function has two objectives: evolving successful as well as undetectable attacks. In particular, the shellcode must contain the following sequence of 'core' components in order to conduct the exploit:

1. Contain open ("/etc/passwd");
2. Contain write ("toor::0:0:root:/root:/bin/bash")[4];
3. Contain close ("/etc/passwd");
4. Execute close after write and open before write;
5. When the system call sequence is fed to Stide, anomaly rate should be as low as possible.

[4] Creates a user 'toor' with super-user privileges, who can connect remotely without supplying a password.

Algorithm 1. Generic Fitness Function

1. Fitness = 0;
   (a) IF ($\{open('/etc/passwd')\} \in$ sequence)THEN (Fitness += 1);
   (b) IF ($\{write('toor :: 0 : 0 : root : /root : /bin/bash')\} \in$ sequence) THEN (Fitness += 1);
   (c) IF ($\{close('/etc/passwd')\} \in$ sequence) THEN (Fitness += 1);
   (d) IF ($'open'$ precedes $'write'$) THEN (Fitness += 1);
   (e) IF ($'write'$ precedes $'close'$) THEN (Fitness += 1);
   (f) $Fitness+ = \frac{100 - Anomaly\ Rate}{20}$

This leads to the final composition of the fitness function, Algorithm 1. A total of five points are awarded for establishing the above components of the 'core' attack. A further five points are awarded for minimizing the anomaly rate provided by the Stide detector. A perfect individual would therefore have a fitness of ten.

3.3 Evolutionary Model

Table 2 defines the instruction set architecture (and parameter types) as per the earlier analysis of application behavior; thus the instruction set consists of the fifteen most frequently occurring system calls characterizing normal behavior, Table 1. Individuals are defined using a fixed length format, with solutions taking the form of sequences of system calls. No registers are required to store state, thus, strictly speaking, this is a Genetic Algorithm as opposed to a (linear) Genetic Program.

Search operators are used independently (children result from any combination of the three operators) and have three forms: crossover, instruction mutation, and instruction swap, Table 3. Crossover takes the form of single point crossover, with the same crossover point utilized in both individuals. The swap operator selects two instructions from the same individual with equal probability and interchanges their respective positions; thus providing the basis to investigate different permutations of the same instructions. In the case of mutation, three forms are investigated.

- *Individual-wise mutation*: selects a single instruction with uniform probability and replaces it with a different instruction from the instruction set, again chosen with uniform probability.
- *Instruction-wise mutation*: tests each instruction independently for the application of the mutation operator. Following a positive test, the instruction is again replaced with another from the instruction set (uniform probability).
- *Greedy mutation*: the current best case individual is selected and a greedy search is performed for the single instruction modification that results in the most improvement to the fitness function. This implies that all 14 alternative instructions are evaluated at each instruction position. Given the computational cost of accessing such an operator the test is only applied every 1,000 tournaments.

In the case of the individual-wise and instruction-wise mutation operators, a linear annealing schedule is employed such that at the last tournament, the mutation probability is zero, decaying linearly with increasing tournament count. The basic motivation being to enable the crossover operator to investigate different contexts of population material as the tournaments advance.

The selection operator takes the form of a steady state tournament, thus the population is inherently elitist with the best individuals always surviving.

Table 2. Instruction Set

System Call	Parameter 1	Parameter 2
open	{"/etc/passwd", "/tmp/dummy"}	n/a
close	{"/etc/passwd", "/tmp/dummy"}	n/a
read	{"/etc/passwd", "/tmp/dummy"}	4 byte space address
write	{"/etc/passwd", "/tmp/dummy"}	{"toor::0:0:root:/root:/bin/bash", "Hello, world!"}
other	n/a	n/a

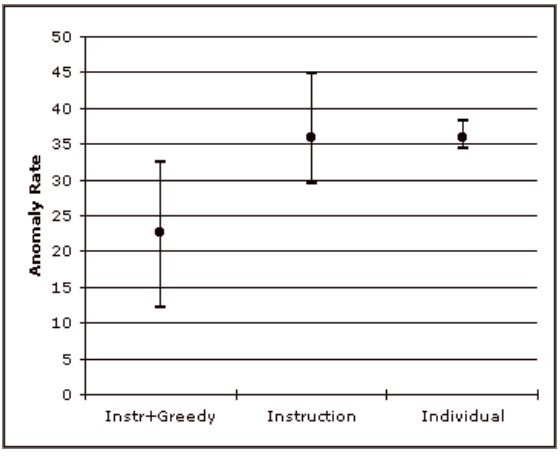
Table 3. Parameters for Evolutionary Search

Parameter	Value
Population	500
Crossover	0.9
Mutation (individual wise)	0.5 with linear decay
Mutation (instruction wise)	0.001 with linear decay
Greedy Mutation	Every 1 000 tournaments
Swap	0.5
Tournament Size	4
Stop Criterion	100 000 tournaments

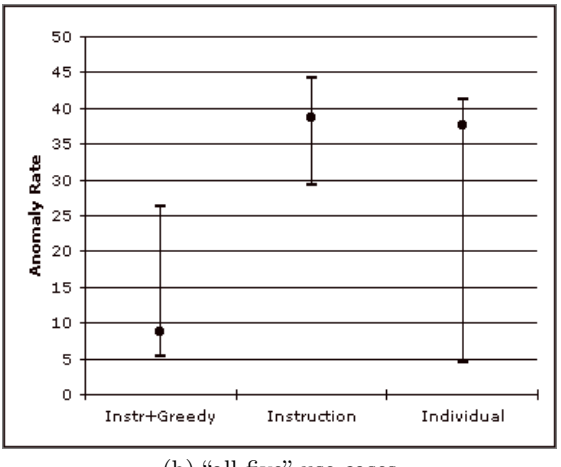
4 Experiments

We begin by establishing the anomaly rate for the original 'core' attack [8] on the two use cases used to configure Stide. This defines the minimum performance for any attack we evolve. Both configurations return an anomaly rate of approximately 67%; thus in order for evolved mimicry attacks to represent an improvement over the original attack, they should return an anomaly rate significantly lower than this.

The principle evolutionary parameter of interest in this work is the significance of the mutation operators employed. To this end, three scenarios are considered of increasing complexity: Individual-wise mutation; Instruction-wise mutation; Instruction-wise mutation with greedy mutation. In all three cases both crossover and swap operators appear, Table 3. Figure 1 details the corresponding percent anomaly rate over 10 runs for Stide configured under scenario 1 (a single use trace)



(a) 'nist' use case



(b) "all five" use cases

Fig. 1. Anomaly rate of attacks evolved against Stide configured over scenario 1 (a) and scenario 2 (b)

and scenario 2 (five use traces) respectively. For completeness we also summarize the anomaly rate of best case attacks, Table 4.

It is immediately apparent that augmenting the search operators with increasingly sophisticated mutation operators results in a direct improvement to the median anomaly rate of the associated evolved exploits. Moreover, it is also apparent that although configuring Stide using a single use case represents a more difficult problem, all attacks returned a lower anomaly rate than the original core attack. The principle difference between the two use case scenarios appear to be manifest in the degree of variation in attack anomaly rates, with the more difficult scenario resulting in a lower variance in anomaly rates. However, there is very little variation in best case attack anomaly rates, with all

Table 4. Percent anomaly rate of best case attacks evolved

Stide Configuration	Instruction-wise and Greedy Mutation	Instruction-wise Mutation	Individual-wise Mutation
all 5 use cases	2.11%	2.97%	3.81%
single 'nist' use case	6.36%	2.97%	5.08%

search operator combinations returning attacks with anomaly rates lower than 4% under Stide configured with five use cases, and less than 6.5% anomaly rate under Stide configured using a single use case.

5 Conclusion

In this work, we developed an evolutionary mimicry attack approach to perform vulnerability testing on the well known Stide host based anomaly detector whilst treating the detector as a black box. That is to say, unlike previous approaches to mimicry attack generation, information from the detector is limited to that available to a "would be" attacker. Specifically, no use is made of privileged data structures internal to the detector. This means that the only feedback employed from the detector during the evolution of attacks is the detector anomaly rate, where this constitutes open information available to users as part of detector deployment. Conversely, previous approaches to mimicry attack generation have concentrated on reverse engineering the normal behavior database from the detector using an exhaustive search [1,2,3,4]. Such an approach would not be feasible without access to privileged detector information, however, the resulting attacks do return an anomaly rate of zero.

A central theme in the approach is the utilization of a Genetic Algorithm to actually automate the process of malicious code design. To do so, a framework is utilized in which specific emphasis is placed on the: (i) Identification of an appropriate set of system calls from which exploits are built, in this case informed by the most frequently executed instructions from the vulnerable application. (ii) Identification of appropriate goals, where these take two basic forms, minimization of detector anomaly rate, whilst matching key steps in establishing the 'core' exploit. (iii) Support for obfuscation, where in this case this is a direct side effect of the stochastic search operators inherent in an evolutionary search. (iv) Search operators benefit from instruction wise mutation and an anealing scheme. Inclusion of a greedy instruction wise mutation operator is also beneficial, but expensive computationally on account of the number of fitness evaluations necessary to resolve a single application of the operator.

Future work will investigate the optimization of search operators for identifying detector blind spots more effectively than is currently the case. Moreover, we are interested in integrating attack evolution into a co-evolutionary context. That is to say, coevolution of attack-detector pairs will enable attacks previously unseen in the environment to be encountered and appropriate responses evolved on a continuous basis. A pre-requisite for such a system, however, requires the

development of an evolutionary detection paradigm based on one-class training. Specifically, in order to avoid the issue of finding an appropriate characterization of normal behavior (an exceptionally difficult task, that typically results in system specific solutions) we recommend the utilization of classifiers trained on attack data alone. Such a class of classifier has been demonstrated using SVMs, where future work will investigate the utility of such a scheme.

Acknowledgments

The authors gratefully acknowledge the support of CFI New Opportunities, NSERC Discovery, and MITACS grants (Canadian government), and SwissCom Innovations Inc. (Switzerland). The first author is a recipient of a Killam predoctoral scholarship.

References

1. D. Wagner and P. Soto, Mimicry attacks on host based intrusion detection systems, ACM Conference on Computer and Communications Security, pp. 255-264, 2002.
2. Tan, K.M.C., Killourhy, K.S., Maxion, R.A., Undermining an Anomaly-based Intrusion Detection System using Common Exploits, RAID'2002, LNCS 2516, pp 54-73, 2002.
3. C. Kruegel, E. Kirda, D. Mutz, W. Robertson, and G. Vigna, Automating mimicry attacks using static binary analysis, Proceedings of the USENIX Security Symposium, pp. 717-738, 2005.
4. K. M. C. Tan, John McHugh, Kevin S. Killourhy, Hiding Intrusions: From the Abnormal to the Normal and Beyond, Symposium on Information Hiding, pp. 1-17, 2002.
5. H.G. Kayacik, A.N. Zincir-Heywood, M.I. Heywood, Evolving Successful Stack Overflow Attacks for Vulnerability Testing, 21st Annual Computer Security Applications Conference, pp. 225-234, 2005.
6. H.G. Kayacik, M.I. Heywood, A.N. Zincir-Heywood. On Evolving Buffer Overflow Attacks using Genetic Programming. Proceedings of the Genetic and Evolutionary Computation Conference, SIGEVO, Volume 2, ACM Press, 1667-1673, July 8-12, 2006.
7. University of New Mexico, Computer Science Department, Computer Immune Systems Data Sets and Software, http://www.cs.unm.edu/~immsec/data-sets.htm, Last accessed May 2006.
8. Securiteam Web Site, Linux Traceroute Exploit Code Released (GDB), Oct 2002, http://www.securiteam.com/exploits/6A00A1F5QM.html, Last accessed May 2006.

Genetic Representations for Evolutionary Minimization of Network Coding Resources

Minkyu Kim[1], Varun Aggarwal[2], Una-May O'Reilly[2],
Muriel Médard[1], and Wonsik Kim[1]

[1] Laboratory for Information and Decision Systems
[2] Computer Science and Artificial Intelligence Laboratory
Massachusetts Institute of Technology, Cambridge, MA 02139, USA
{minkyu@, varun_ag@, unamay@csail., medard@, wskim14@}mit.edu

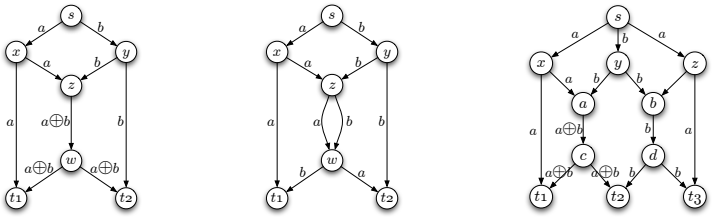
Abstract. We demonstrate how a genetic algorithm solves the problem of minimizing the resources used for network coding, subject to a throughput constraint, in a multicast scenario. A genetic algorithm avoids the computational complexity that makes the problem NP-hard and, for our experiments, greatly improves on sub-optimal solutions of established methods. We compare two different genotype encodings, which tradeoff search space size with fitness landscape, as well as the associated genetic operators. Our finding favors a smaller encoding despite its fewer intermediate solutions and demonstrates the impact of the modularity enforced by genetic operators on the performance of the algorithm.

1 Introduction

Network coding is a novel technique that generalizes routing. In traditional routing, each interior network node, which is not a source or sink node, simply forwards the received data or sends out multiple copies of it. In contrast, network coding allows interior network nodes to perform arbitrary mathematical operations, e.g., summation or subtraction, to combine the data received from different links. It is well known that network throughput can be significantly increased by network coding [1, 2, 3]. While network coding is assumed to be done at all possible nodes in most of the network coding literature, it is often the case that network coding is required only at a subset of nodes to achieve the desired throughput. Consider Example 1:

Example 1. In the canonical example of network B (Fig. 1(a)) [1], where each link has unit capacity, source s can send 2 units of data simultaneously to the sinks t_1 and t_2, which is not possible with routing alone. But only node z needs to combine its two inputs while all other nodes perform routing only. If we suppose that link (z, w) in network B has capacity 2, which we represent by two parallel unit-capacity links in network B' (Fig. 1(b)), a multicast of rate 2 is possible without network coding. In network C (Fig. 1(c)), where node s is to transmit data at rate 2 to the 3 leaf nodes, network coding is required either at node a or at node b, but not at both. □

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 21–31, 2007.
© Springer-Verlag Berlin Heidelberg 2007



(a) Network B (b) Network B' (c) Network C

Fig. 1. Sample networks for Example 1

Example 1 leads us to the following question: To achieve the desired throughput, at which nodes does network coding need to occur? This question's answer is valuable because eliminating unnecessary coding nodes will save computation at the application layer if that is where network coding is handled. Alternatively, if network coding is integrated in the buffer management of routers, it will reduce the number of routers that need to perform coding operations without compromising communication capacity. For a GA, the problem can be posed as the minimization of coding cost (in links or nodes) subject to the constraint of feasibility (achieving the desired throughput).

The problem of determining a minimal set of nodes where coding is required is NP-hard; its decision problem, which decides whether the given multicast rate is achievable without coding, reduces to a multiple Steiner subgraph problem, which is NP-hard [4]. It is shown that even approximating the minimum number of coding nodes within any multiplicative factor or within an additive factor of $|V|^{1-\epsilon}$ is NP-hard [5]. Note, however, that once the set of coding nodes is identified, a network code achieving the desired throughput can be efficiently constructed for the multicast scenario, either in a deterministic [6] or randomized fashion [7].

In the network research community, [8] and [9] have documented results that demonstrate the benefit of the GA over other existing approaches in terms of reducing the number of coding links or nodes and its applicability to a variety of generalized scenarios. These contributions emphasized the computational "how-to" aspects of feasibility checking and the transformations of the network graph into secondary graphs that express possible coding situations uniformly since these are key to evaluating the fitness function of the GA.

In the course of investigating the feasibility checks, graph transformations, and value of using a GA, we experimented with two different genotype encodings[1] and associated operators. For both encodings, we use a genotype composed of a number of blocks, each of which consists of a set of variables indicating the link states. For a block of length k, using an alphabet of cardinality 2, the *Binary Link State* (BLS) encoding represents all possible 2^k states of k links. On the other hand, for the *Block Transmission State* (BTS) encoding, we group

[1] To minimize confusion, throughout the paper, the term "encoding" refers to "genotype encoding" only, while the term "coding" means "network coding."

those link states into $(k + 2)$ transmission states. Despite the smaller search space size of BTS encoding, it is not clear that it should be superior to BLS encoding because in grouping many link states into one, less information that would relate the fitness of solutions *intermediate* to the best solution is available in contrast to BLS encoding which provides more information through its intermediate solutions.

In this paper we focus on the two different encodings with associated genetic operators and conduct a more comprehensive comparison between them. Specifically relevant to the GA community, we consider into the GA encoding tradeoff issues related to search space size and fitness landscape. The rest of the paper is organized as follows. Section 2 presents the problem formulation, and Section 3 describes the network coding GA (NCGA) with the two different encodings and associated operators. Section 4 sets up a set of experiments into relative values of the encodings and discusses the results. Section 5 presents a summary of the results and our conclusions.

2 Problem Formulation

We assume that a network is given by a directed multigraph $G = (V, E)$ as in [10] where each link has a unit capacity whose unit can be arbitrarily chosen, e.g., k bits per second for a constant k, or a fixed size packet per unit time, etc. Links with larger capacities are represented by multiple links. Only integer flows are allowed, hence there is either no flow or a unit rate of flow on each link. We consider the single multicast scenario in which a single source $s \in V$ wishes to transmit data at rate R to a set $T \subset V$ of sink nodes. Rate R is said to be achievable if there exists a transmission scheme that enables all $|T|$ sinks to receive all of the information sent. We only consider linear coding, where a node's output on an outgoing link is a linear combination of the inputs from its incoming links. Linear coding is sufficient for multicast [2].

Given an achievable rate R, we wish to determine a minimal set of nodes where coding is required in order to achieve this rate. However, whether coding is necessary at a node is determined by whether coding is necessary at at least one of the node's outgoing links and thus, as pointed out also in [5], the number of coding links is in fact a more accurate estimator of the amount of computation incurred by coding. We assume hereafter that our objective is to minimize the number of coding *links* rather than *nodes*. Note, however, that as demonstrated in [8], it is straightforward to generalize the proposed algorithm to the case of minimizing the number of coding nodes. Furthermore, [8] shows that, with appropriate changes, the algorithm can be readily applied to more generalized optimization scenarios, e.g., when links and nodes have different coding costs.

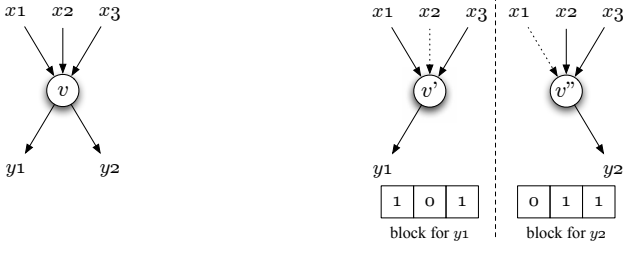
It is clear that no coding is required at a node with only a single input since these nodes have nothing to combine with [8]. For a node with multiple incoming links, which we refer to as a *merging node*, if the linearly coded output to a particular outgoing link weights all but one incoming message by zero, effectively no coding occurs on that link; even if the only nonzero coefficient is not

identity, there is another coding scheme that replaces the coefficient by identity [5]. Thus, to determine whether coding is necessary at an outgoing link of a merging node, we need to verify whether we can constrain the output of the link to depend on a single input without destroying the achievability of the given rate. As in network C of Example 1, the necessity of coding at a link depends on which other links code and thus the problem of deciding where to perform network coding in general involves a selection out of exponentially many possible choices. We employ a GA-based search method to efficiently address the large and exponentially scaling size of the space.

3 Network Coding GA (NCGA)

Prior to using the NCGA, the given network graph G is transformed into a secondary graph by either of the two methods presented in [8, 9][2]. Regardless which method is used, mapping the network coding problem to a GA framework is done as follows.

Suppose a merging node with $k(\geq 2)$ incoming links. To consider the transmission to *each* of its outgoing links, we assign a binary variable to each of its k incoming links, which being 1 indicates that the link state is *active* (the input from the associated incoming link is transmitted to the outgoing link) and 0 indicates it is *inactive*. Given that network coding is required for the transmission only if two or more link states are active, we may need to consider those k variables together. We refer to the set of the k variables as a *block* of length k (see Fig. 2 for an example). The way how those binary variables are actually encoded as a genotype will be described later in this section.



(a) Merging node v (b) Two blocks for outgoing links of v

Fig. 2. Node v with 3 incoming and 2 outgoing links results in 2 blocks, each with 3 variables indicating the states of incoming links (x_1, x_2, x_3) onto the associated outgoing link

Constraint and Fitness Function. A genotype is called *feasible* if there exists a network coding scheme that achieves the given rate R with the link states

[2] An interested reader is referred to [8, 9] for the details of the two methods for graph transformation and feasibility testing.

determined by the genotype. To calculate the fitness of genotype $\underline{y}$, its feasibility must be checked by either of the two methods in [8], [9] depending on the secondary graph chosen earlier, and the fitness value F is assigned as

$$F(\underline{y}) = \begin{cases} \text{number of blocks with two or more active links,} & \text{if } \underline{y} \text{ is feasible,} \\ \infty, & \text{if } \underline{y} \text{ is infeasible.} \end{cases}$$

The NCGA uses a standard generation-based GA control loop with tournament selection. It terminates at some maximum number of generations. Afterward, the best solution of the run is optimized with *greedy sweep*: we switch each of the remaining 1's to 0 if it can be done without violating feasibility. This procedure may only improve the solution, and sometimes the improvement can be substantial. Reference [9] proves that the NCGA with greedy sweep is guaranteed to perform no worse than the existing algorithm in [5].

Binary Link State (BLS) Encoding and Operators. This encoding allows a block of length k to take any of 2^k possible binary strings of length k. If we denote by d_{in}^v and d_{out}^v the in-degree and the out-degree of node v, node v has d_{out}^v blocks of length d_{in}^v, and we have a total of $m = \sum_{v \in \mathcal{V}} d_{in}^v d_{out}^v$ binary variables, where $\mathcal{V}$ is the set of all merging nodes. We must explore the m-dimensional binary space of 2^m candidates to find the desired minimal set of coding links.

For BLS encoding we use uniform crossover, where each pair of genotypes is selected for crossover with a given probability (mixing ratio) and the two genotypes in a selected pair exchange each bit independently with another given probability (crossover probability). For mutation, we use simple binary mutation, where each bit in each genotype is flipped independently with a given probability (mutation rate). Since these operators deal with each bit separately, we refer to the operators used for BLS encoding as *bit-wise* genetic operators.

Block Transmission State (BTS) Encoding and Operators. As mentioned above, once a block has at least two 1's, replacing all the remaining 0's with 1's has no effect on whether coding is done. Moreover, it can be shown that substituting 0 with 1, as opposed to substituting 1 with 0, does not hurt the feasibility. Therefore, for a feasible genotype, any block with two or more 1's can be treated the same as the block with all 1's. Thus we could group all the states with two or more active links into a single state, *coded* transmission. This state is rounded out by k states for the *uncoded* transmissions of the input received from one of the k single incoming links and one state indicating *no* transmission. Thus BTS encoding emerges where each block of length k can only take one of the following $(k+2)$ strings: "111...1", "100...0", "010...0", "001...0", ..., "000...1", "000...0". The net effect is a reduction in the number of possible states for a block to $(k+2)$ rather than 2^k. If we let w be the total number of blocks (i.e., $w = \sum_{v \in \mathcal{V}} d_{out}^v$) and k_i denote the length of the i-th block ($i = 1, ..., w$), the search space size is $\prod_{i=1}^{w}(k_i + 2)$. However, the benefit of the smaller space size in fact comes at the price of losing the information on the partially active link states that may serve as intermediate steps toward an uncoded transmission state. This tradeoff will be discussed more in depth in Section 4.

To preserve the BTS encoding structure throughout genetic operations, we need to define a new set of genetic operators, which we refer to as *block-wise* genetic operators. For block-wise uniform crossover, we let two genotypes subject to crossover exchange each block, rather than bit, independently with the given crossover probability. For block-wise mutation, we let each block under mutation take another string chosen uniformly at random out of $(k+1)$ other strings for a length-k block. If mutation rate is α, the average number of changed bits in a length-k block is now calculated as $\frac{4k^2}{(k+1)(k+2)}\alpha$, whereas it is $k\alpha$ for bit-wise mutation. While the difference becomes more apparent when k is large since the latter is upper bounded by 4α, those values are still different for $k = 2$, where the block structure makes no difference in the space size. Though block-wise mutation may lead to much smaller number of flipped bits, it is more likely to cause a sudden change in a genotype.

4 Experiments

4.1 Experiment Setup

The two encodings not only differ in the size of search space, but in the way the genetic operators are applied. In BTS encoding with the block-wise operators, crossover is applied at the block boundaries and mutation is performed intra-block. However, in BLS encoding with the bit-wise operators, crossover and mutation are randomly applied without respecting any block boundaries. While evaluating the effect of the search space size reduction, we also want to investigate whether the exploitation of block level modularity by the block-wise operators gives any significant improvement in the algorithm's performance. We thus set up two experiments: Experiment I compares the effect of the two encodings combined with associated operators on the performance of the NCGA, while Experiment II tests the effect of the operators alone by isolating the effect of the encodings that lead to different space sizes.

Experiment I: We use two acyclic networks, I-50 and I-75, generated by the algorithm in [11], whose details are given in Table 1. Note that BTS encoding reduces the size of the search space by 30.3 and 115 orders of magnitude for networks I-50 and I-75, respectively, compared with that in the case of BLS encoding. This experiment tests which encoding is better given the tradeoff in the search space size and ease of traversing the fitness landscape.

Experiment II: We construct a set of synthetic networks with only blocks of length 2. Note that for a block of length 2, the two encodings have the same search space size ($2^k = k + 2$ when $k = 2$), but the block-wise operators retain their modularity. These networks are constructed by cascading a number of copies of network B' in Fig. 1(b) such that the source of each subsequent copy of B' is replaced by an earlier copy's sink. We use fixed-depth binary trees containing 3, 7, 15, and 31 copies of B' (henceforth called II-3, II-7, II-15, and II-31, respectively). Parameters of these networks are given in Table 1. All these

network have 0 as the minimum number of coding links, i.e., multicast rate 2 is achievable without coding. We scale up the network size to investigate the payoff one gets with modular operators as the search space size increases.

We use the NCGA with the decomposition-based graph transformation and the max-flow feasibility testing described in [9]. For comparison, we also perform experiments using the two existing approaches by Fragouli et al. [12] ("Minimal 1")[3], and Langberg et al. [5] ("Minimal 2"), in both of which link removal is done in a random order. For Minimal 1, the subgraph is selected also by a minimal approach, which starting from the original graph sequentially removes the links whose removal does not destroy the achievability.

Table 1. Details of the networks used in the experiments

Network	Genotype length	Number of blocks	Avg. length of blocks	Search space size ($\log_{10}$) (BLS/BTS)
I-50	280	71	3.94	84.29/53.93
I-75	761	130	5.85	229.08/113.47
II-3	32	16	2	9.63/9.63
II-7	80	40	2	24.08/24.08
II-15	176	88	2	52.98/52.98
II-31	368	184	2	110.78/110.78

4.2 Algorithm Parameters

We set the total budget of fitness evaluations to 150,000 (a very small fraction of the search space size of the networks considered. Preliminary experiments suggested different tournament sizes and mutation rates for the two encodings: 10 and 0.006 for BLS encoding, and 100 and 0.012 for BTS encoding, respectively. All other parameters are matched for the two encodings. We perform 30 runs for each network with both encodings. Table 2 summarizes the parameters used.

Table 2. GA parameters used in the experiments

Population size	150
Tournament size	10(BLS)/100(BTS)
Maximum generations	1000
Mixing ratio/Crossover probability	0.8/0.8
Mutation rate	0.006(BLS)/0.012(BTS)

When randomly initializing the population, we insert an all-one vector, which represents the solution where coding is done everywhere and thus is feasible by the assumption that the given rate R is achievable. The role of the all-one vector

[3] Though minimizing network coding resources is not its main concern, [12] presents an algorithm to obtain a subgraph with a minimal number of coding links.

as a feasible starting point is crucial to the performance of the algorithm as discussed in [8].

4.3 Experimental Results

Results for the both experiments are summarized in Table 3. The table shows the optimal fitness achieved by each algorithm averaged over 30 runs. The statistical significance of the difference between BTS and BLS encodings is measured by conducting paired t-tests and the p-values are reported in the last row of the table.

Table 3. Performance of the algorithms for each network. Each value in brackets is standard deviation.

	I-50	I-75	II-3	II-7	II-15	II-31
NCGA/BLS	3.33(1.03)	6.43(1.30)	0.93(0.69)	2.20(1.27)	5.57(1.55)	12.43(2.37)
(w/o greedy sweep)	3.33(1.03)	39.93(2.74)	0.93(0.69)	2.20(1.27)	5.57(1.55)	12.43(2.37)
NCGA/BTS	2.40(0.62)	3.63(0.61)	0.00(0.00)	0.00(0.00)	0.17(0.38)	1.03(0.81)
(w/o greedy sweep)	2.40(0.62)	3.63(0.61)	0.00(0.00)	0.00(0.00)	0.17(0.38)	1.07(0.83)
Minimal 1	4.90(1.37)	9.50(2.16)	3.00(0.00)	7.00(0.00)	15.50(0.00)	31.00(0.00)
Minimal 2	4.33(1.37)	7.90(1.71)	2.13(0.86)	4.37(1.25)	9.90(1.65)	19.97(2.66)
p-value	$8.21e^{-5}$	$2.65e^{-15}$	$6.7e^{-10}$	$2.2e^{-13}$	$2.9e^{-26}$	$1.55e^{-32}$

Experiment I: For both networks I-50 and I-75, the NCGA with greedy sweep, with either of the two encodings, outperforms the two existing minimal approaches. Between the two encodings, BTS encoding gives rise to a substantial performance gain over BLS encoding with the statistical significance confirmed by the tabulated p-values.

Experiment II: Again the NCGA with BTS encoding outperforms that with BLS encoding on average for all networks, while either of the two performs significantly better than the minimal algorithms. For networks II-3 and II-7, the NCGA with BTS encoding finds the optimal (0 coding links) in all of the 30 runs. For networks II-15 and II-31, it succeeds to find the optimal solution 25 and 8 times, respectively. On the other hand, BLS encoding does not find the optimal number of coding links in any of the 30 runs for networks II-7, II-15, and II-31.

The average performance of the NCGA with both encodings is plotted against the logarithm of the search space size in Fig. 3. The plot suggests a linear scaling of algorithms as the search space size grows exponentially. More data points would lend more confidence to this hypothesis. The curve for BTS encoding has a much smaller intercept and slope than BLS encoding, implying that the payoff of the block-wise operators increases as the search space size increases.

4.4 Discussion of Results

Experiment I clearly indicates that BTS encoding is better than BLS encoding for the networks considered. We can thus conclude that the benefits of the smaller

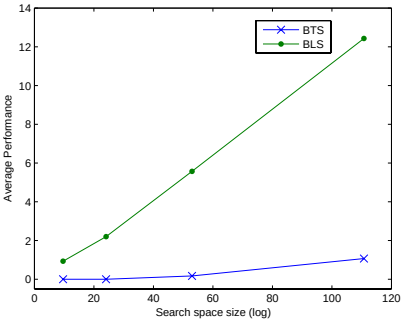


Fig. 3. Average performance of NCGAs with log of search space size

search space trump the challenge of the more difficult fitness landscape. For network I-50, BTS encoding improves over BLS encoding on average by a single coding link. Though small, this difference is statistically significant. For network I-75, without greedy sweep, the average difference in performance between the two algorithms is much higher, i.e., 34 coding links. This large difference in performance can be attributed to two specific factors: the much larger search space size (see Table 1) and larger average block size. The difference also indicates that the information on the intermediate solutions that BLS encoding provides may not be particularly useful without guaranteeing that those intermediate steps ultimately lead to an uncoded transmission state.

Experiment II demonstrates the superiority, by a remarkably large margin, of the block-wise operators over the bit-wise operators. It also indicates that both NCGAs scale linearly with an exponentially growing search space size (see Fig. 3), which is remarkable. This prompts due analysis of the difference between the two operators. When applied to the pair of blocks "00" and "11", the block-wise crossover cannot result in either block "01" or "10". However, for the bit-wise crossover, the pair of blocks "00" and "11" may result in "00", "01", "10", or "11". It can be shown that with probability $\frac{1}{4}$ the two crossovers behave differently, if the population has equal frequency of all block types. Let us recall that the block-wise mutation leads to a smaller number of changed bits on average than the bit-wise mutation (Section 3). Nevertheless, the block-wise mutation exhibits higher "exploratory power" than the bit-wise mutation in the sense that it is more likely to lead to changes in multiple bits. For the block-wise mutation, given any block, the remaining three blocks are equally likely to occur on mutation. Thus, if mutation rate is α, the probabilities of 0, 1, 2-bit change are $1-\alpha$, $\frac{2}{3}\alpha$, $\frac{1}{3}\alpha$, respectively, whereas those probabilities in the bit-wise case are $(1-\alpha)^2$, $2\alpha(1-\alpha)$, α^2, respectively. Provided that $\alpha < \frac{1}{3}$, the probability of 2-bit change is larger for the block-wise mutation. A similar analysis can be done for the whole genotype as well.

One may speculate that the better performance of the block-wise operators is due to the higher exploratory power of the block-wise mutation rather than the modularity of the operators. To confirm the contrary, we consider a new set of

Table 4. Performance of NCGA with MHD operators. Refer to Table 3 for comparison with NCGAs with bit-wise or block-wise operators.

	II-3	II-7	II-15	II-31
NCGA/MHD	0.77(0.68)	2.47(1.33)	5.83(1.68)	12.63(3.23)

operators, called the *Matched Hamming Distance* (MHD) operators, where the MHD mutation leads to the statistically same Hamming distance changes as the block-wise mutation, but exhibits no positional bias as to where the mutation is applied, and the MHD crossover is the same as the bit-wise crossover which neither imposes modularity. From Table 4 compared with Table 3, we observe that the MHD operators perform similarly as the bit-wise operators, but far worse than the block-wise operators. We can thus confidently claim that the respect for modularity enforced by the block-wise operators is the main cause of the superior performance of the block-wise operators.

5 Conclusions and Future Work

For our suite of network coding problems, we have found that the benefits of the smaller search space and modular operators trump the challenge of the more difficult fitness landscape. In the future, we will study the effect of exploiting further modularity with BTS operators that cross over at merging node boundaries and perform intra-block mutations. We could incorporate hierarchical modularity using domain knowledge of sparsely connected regions, regions with similar structure or simply neighboring nodes. A hierarchical structure of crossover boundaries could be formed and applied with different probabilities. These results will inform the GA community and help push the state-of-art in algorithms for minimizing network coding resources.

References

1. Ahlswede, R., Cai, N., Li, S.Y.R., Yeung, R.W.: Network information flow. IEEE Trans. Inform. Theory **46**(4) (2000) 1204–1216
2. Li, S.Y.R., Yeung, R.W., Cai, N.: Linear network coding. IEEE Trans. Inform. Theory **49**(2) (2003) 371–381
3. Fragouli, C., Le Boudec, J.Y., Widmer, J.: Network coding: An instant primer. SIGCOMM Comput. Commun. Rev. **36**(1) (2006) 63–68
4. Richey, M.B., Parker, R.G.: On multiple Steiner subgraph problems. Networks **16**(4) (1986) 423–438
5. Langberg, M., Sprintson, A., Bruck, J.: The encoding complexity of network coding. In: Proc. IEEE ISIT. (2005)
6. Jaggi, S., Sanders, P., Chou, P.A., Effros, M., Egner, S., Jain, K., Tolhuizen, L.: Polynomial time algorithms for multicast network code construction. IEEE Trans. Inform. Theory **51**(6) (2005) 1973–1982
7. Ho, T., Koetter, R., Médard, M., Karger, D.R., Effros, M.: The benefits of coding over routing in a randomized setting. In: Proc. IEEE ISIT. (2003)

8. Kim, M., Ahn, C.W., Médard, M., Effros, M.: On minimizing network coding resources: An evolutionary approach. In: Proc. NetCod. (2006)
9. Kim, M., Médard, M., Aggarwal, V., O'Reilly, U.M., Kim, W., Ahn, C.W., Effros, M.: Evolutionary approaches to minimizing network coding resources. In: Proc. IEEE Infocom (to appear). (2007)
10. Koetter, R., Médard, M.: An algebraic approach to network coding. IEEE/ACM Trans. Networking **11**(5) (2003) 782–795
11. Melançon, G., Philippe, F.: Generating connected acyclic digraphs uniformly at random. Inf. Process. Lett. **90**(4) (2004) 209–213
12. Fragouli, C., Soljanin, E.: Information flow decomposition for network coding. IEEE Trans. Inform. Theory **52**(3) (2006) 829–848

Bacterial Foraging Algorithm with Varying Population for Optimal Power Flow

M.S. Li[1], W.J. Tang[1], W.H. Tang[1], Q.H. Wu[1,⋆], and J.R. Saunders[2]

[1] Department of Electrical Engineering and Electronics
qhwu@liv.ac.uk
[2] School of Biological Sciences
The University of Liverpool, Liverpool L69 3GJ, U.K.

Abstract. This paper proposes a novel optimization algorithm, Bacterial Foraging Algorithm with Varying Population (BFAVP), to solve Optimal Power Flow (OPF) problems. Most of the conventional Evolutionary Algorithms (EAs) are based on fixed population evaluation, which does not achieve the full potential of effective search. In this paper, a varying population algorithm is developed from the study of bacterial foraging behavior. This algorithm, for the first time, explores the underlying mechanisms of bacterial chemotaxis, quorum sensing and proliferation, etc., which have been successfully merged into the varying-population frame. The BFAVP algorithm has been applied to the OPF problem and it has been evaluated by simulation studies, which were undertaken on an IEEE 30-bus test system, in comparison with a Particle Swarm Optimizer (PSO) [1].

Keywords: Evolutionary algorithm, Bacterial foraging algorithm, Varying population, Optimal power flow.

1 Introduction

Optimal Power Flow (OPF) problem has been intensively studied as a network constrained economic dispatch problem, since its introduction by Carpentier [2] in 1962. Generally, the OPF problem aims to achieve the minimization of a specific objective function from a model of a power system, by adjusting the control variables of the system, while satisfying a set of operational and physical constraints. As a result, the OPF problem can be formulated as a non-linear constrained optimization problem.

Various conventional optimization methods have been developed to solve the OPF problem in the past a few decades, such as Nonlinear Programming (NLP) [3] [4], Quadratic Programming (QP) [5], Linear Programming (LP) [6], and interior point methods [7] [8]. However, these traditional methods do not guarantee finding the global optimum solution, since they more or less rely on the convexity of the objective functions, which would not be always satisfied in many practical

⋆ Corresponding author.

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 32–41, 2007.
© Springer-Verlag Berlin Heidelberg 2007

cases. In most cases, the traditional methods are suitable for single-peak and linear objective functions. To tackle this problem, EAs, especially PSO, have been applied to solve the OPF problem [9]. These algorithms are all based on fixed population evaluation, which limits their computational capability and introduces redundant computation in the optimization process. Therefore, a varying population algorithm is proposed to overcome the above drawbacks.

In this paper, a Bacterial Foraging Algorithm with Varying Population (BFAVP) is proposed for solving the OPF problem. This work is based on the Bacterial Foraging Algorithm (BFA) [10], which is inspired by *E.coli* chemotactic behavior. However, instead of simply describing chemotactic behavior into the algorithm, BFAVP also incorporates the mechanisms of bacterial proliferation and quorum sensing, which allow a varying population in each generation of bacterial foraging process. The BFAVP has been evaluated on a practical OPF problem, focusing on minimizing the fuel cost of a power system, in comparison with PSO. The evaluation of the algorithm was carried out using an IEEE 30-bus power system. The simulation results are reported in this paper to show the merits of the proposed algorithm.

The paper is organized as follows: Section 2 introduces the formulation of OPF problem. The detailed description of BFAVP, including bacterial chemotaxis, proliferation and its influence to the environment, and a simplified quorum sensing mechanism, are presented in Section 3. The simulation studies are undertaken on an IEEE 30-bus test system. The results of BFAVP in solving the OPF problem are given in Section 4, followed by the conclusion in Section 5.

2 Optimal Power Flow Problem Formulation

For an OPF problem, the control variables include the generator real power, the generator bus voltages, the tap ratios of transformers and the reactive power generation of var sources. The state variables are the slack bus power, the load bus voltage, the generator reactive power output, and the network power flows. The constrains include inequality ones which are the limits of the control variables and state variables; and equality ones which are the power flow equations. The OPF problem can be formulated as a constrained optimization problem as follows:

$$\min \quad f(x,u) \tag{1}$$

$$\text{s.t.} \quad g(x,u) = 0 \tag{2}$$

$$h(x,u) = 0 \tag{3}$$

where f is the optimization objective function, g is a set of constrain equations, and h is a set of formulated constrain inequations. x is the vector of dependent variables such as the slack bus power P_{G_1}, the load bus voltage V_L, generator reactive power outputs Q_G and the apparent power flow S_k. x can be expressed as:

$$x^T = [P_{G_1}, V_{L_1} ... V_{L_{N_G}}, Q_{G_1} ... Q_{G_{N_G}}, S_1 ... S_{N_E}]. \tag{4}$$

u is a set of the control variables such as the generator real power output P_G expect at the slack bus P_{G_1}, the generator voltages V_G, the transformer tap

setting T, and the reactive power generations of var source Q_C. Therefore, u can be expressed as:

$$u^T = [P_{G_2}...P_{G_{N_G}}, V_{G_1}...V_{G_{N_G}}, T_1...T_{N_T}, Q_{C1}...Q_{C_{N_C}}]. \tag{5}$$

The equality constrains $g(x, u)$ are the nonlinear power flow equations which are formulated as below:

$$0 = P_{G_i} - P_{D_i} - V_i \sum_{j \in N_i} V_j (G_{ij} \cos\theta_{ij} + B_{ij} \sin\theta_{ij}) \quad i \in N_0 \tag{6}$$

$$0 = Q_{G_i} - Q_{D_i} - V_i \sum_{j \in N_i} V_j (G_{ij} \sin\theta_{ij} + B_{ij} \cos\theta_{ij}) \quad i \in N_{PQ}. \tag{7}$$

And the inequality constraints $h(x, u)$ are limits of control variables and state variables which can be formulated as:

$$\begin{aligned}
&P_{G_i}^{\min} \le P_{G_i} \le P_{G_i}^{\max} \quad i \in N_G \\
&Q_{G_i}^{\min} \le Q_{G_i} \le Q_{G_i}^{\max} \quad i \in N_G \\
&Q_{C_i}^{\min} \le Q_{C_i} \le Q_{C_i}^{\max} \quad i \in N_C \\
&T_k^{\min} \le T_k \le T_k^{\max} \quad k \in N_T \\
&V_i^{\min} \le V_i \le V_i^{\max} \quad i \in N_B \\
&|S_k| \le S_k^{\max} \quad k \in N_E
\end{aligned} \tag{8}$$

To solve non-linear constrained optimization problems, the most common method uses penalty function to transform a constrained optimization problem into an unconstrained one. The objective function is generalized as follows:

$$\begin{aligned}
F = f + \sum_{i \in N_V^{\lim}} \lambda_{V_i} (V_i - V_i^{\lim})^2 + \sum_{i \in N_Q^{\lim}} \lambda_{G_i} (Q_i - Q_{G_i}^{\lim})^2 \\
+ \sum_{i \in N_E^{\lim}} \lambda_{S_i} (|S_i| - S_i^{\max})^2
\end{aligned} \tag{9}$$

where λ_{V_i}, λ_{G_i}, and λ_{S_i} are the penalty factors. $V_i^{\lim}$ and $Q_{G_i}^{\lim}$ are defined as:

$$V_i^{\lim} = \begin{cases} V_i^{\max} & \text{if} \quad V_i > V_i^{\max} \\ V_i^{\min} & \text{if} \quad V_i < V_i^{\min} \end{cases} \tag{10}$$

$$Q_{G_i}^{\lim} = \begin{cases} Q_{G_i}^{\max} & \text{if} \quad Q_{G_i} > Q_{G_i}^{\max} \\ Q_{G_i}^{\min} & \text{if} \quad Q_{G_i} < Q_{G_i}^{\min} \end{cases} \tag{11}$$

3 Bacterial Foraging Algorithm with Varying Population

Based on bacterial behaviors, a BFAVP model comprises of the following four aspects, *i.e.* chemotaxis, metabolism energy and nutrition losing, proliferation and elimination, and a simplified quorum sensing. Chemotaxis is the basic search

principle of BFAVP. The metabolism energy and nutrition losing phenomenon lead to the bacterial proliferation and elimination, which cause the variation population sizes. By using the above features, BFAVP can be used to tackle multi-peak, non-linear OPF problems.

3.1 Chemotaxis

Bacteria swim by rotating thin, helical filaments known as flagella driven by a reversible motor embedded in the cell wall. Peritrichously flagellated bacteria such as *E.coli* have 8-10 flagella placed randomly on the cell body [11]. Swimming was found to consist of smooth "runs" interrupted roughly every second by transient "tumble". Chemotaxis is the ability of the cells to move toward distant sources of food molecules, and it is based on the suppression of tumbles in cells. Based on the bacterial chemotactic behavior, BFA has been proposed by Passino in 2002 [10]. In BFA, nourishment environment stands for an optimization objective function, and the BFA algorithm imitates bacterial swimming motions.

In order to adjust the length of a unit tumble, a dynamic parameter $c(t)$ is introduced into BFAVP. $c(t)$ is the length of the step taken in the random direction specified by the tumble in the tth bacterial foraging process.

Let $\theta_i(t_c)$ be the location of the ith bacterium at the t_cth chemotactic step, then the updated position of the next movement after a tumble is:

$$\theta_i(t_c + 1) = \theta_i(t_c) + c(t)\frac{\Delta_i}{\sqrt{\Delta_i^T \Delta_i}} \tag{12}$$

where Δ_i indicates the ith step tumble angle, which can be expressed as:

$$\Delta_i = [\Delta_{i1}, \Delta_{i2}, ..., \Delta_{in}] \tag{13}$$

where n denotes the size of dimensions.

The length of $c(t)$ can be expressed as:

$$c(t) = \frac{\frac{\sum_{i=1}^{p}(E_i(t_c))}{p} - \min\{E(t_c)\}}{\max\{E(t_c)\} - \min\{E(t_c)\}} \tag{14}$$

where $E(t_c)$ is the set of fitness values at the t_cth chemotactic step, which can be expressed as:

$$E(t_c) = \{E_1(t_c), E_2(t_c), ..., E_p(t_c)\} \tag{15}$$

where p is the population size of current generation.

In equation (14), $c(t)$ is a positive number which in the range of (0,1]. In the experiment, $c(t)$ decreases during the optimization. Thus, the dynamic $c(t)$ accelerates the speed of convergence, and increases the convergence accuracy.

3.2 Metabolism Energy and Nutrition Losing

Bacterial metabolism can be broadly divided on the basis of energy usage for chemotaxis, growth and sensing. In BFAVP, the bacterial energy is described as

$e_i(t_c)$, which is a measure of the energy quantity. $e_i(t_c)$ indicates the energy of the ith bacterium at the t_cth chemotaxis step. The evaluation value of $E_i(t_c)$ is the source for the bacterial energy.

For the algorithm implemented, the record of nutrition losing is updated at each evaluation. At each evaluation step, the quantity of nutrition losing record has been subtracted from the fitness value, *i.e.*, the value of optimum decreases when bacteria sink into that position. If there is not enough nutrition in that position, the bacteria have a tendency to swim to a forane space. Pre-mature results are then prevented with this method. After h iterations of the function evaluated, the geometrical center of each nutrition losing area can be expressed as:

$$G = \{g_m = [g_{1m}, g_{2m}, ..., g_{nm}] \mid m = 1, 2, ..., h\} \tag{16}$$

where g_m indicates the mth geometrical center of nutrition losing area, and g_{nm} indicates the mth geometrical center on nth dimension.

Set $X_i = [x_{1i}, x_{2i}, ..., x_{ni}]$ as the coordinate of ith bacterium, and assume bacterium i falls into the geometrical area m, then the distance between the bacterium and the geometrical center is less than the radius of that area. This can be expressed as:

$$|X_i - g_m| < \varepsilon \tag{17}$$

where ε is normalized radius of each area. When the ith bacterium falls into several overlaps of the geometrical area, the energy incremental quantity is:

$$\Delta e_i(t_c) = s \cdot e_b \tag{18}$$

where s denotes the numbers of the geometrical area, and e_b represents the unit quantity of energy. The energy transferred from the environment to a bacterium for metabolism is denoted as $\Delta e_i(t_c)$. Thus, for a bacterium, the updated energy after chemotactic step t_c is defined as:

$$\begin{aligned} e_i(t_c + 1) &= e_i(t_c) + \Delta e_i(t_c) \\ &= e_i(t_c) + s \cdot e_b \end{aligned} \tag{19}$$

Due to the nutrition losing, $s \cdot e_b$ is subtracted from the evaluation value for the bacterium i.

3.3 Proliferation and Elimination

All bacteria reproduce through binary fission, which results in cell division. Two identical clone daughter cells are produced after the cell division. In BFAVP, the reproduction process is controlled by the bacterial energy $e_i(t_c)$; and the elimination is determined by the bacterial foraging process t_c. When $e_i(t_c)$ approaches to the upper limit of the energy boundary, the bacteria will turn into reproduction states.

For the new bacterium, the bacterial energy is:

$$e_{p+1}(t_c + 1) = \frac{e_i(t_c)}{2} \tag{20}$$

where p is the population size in the current state. The previous ith bacterium also keeps half of the energy:

$$e_i(t_c + 1) = \frac{e_i(t_c)}{2} \tag{21}$$

In BFAVP, a counter is set to record the bacterial age for each bacterium. After the evaluation process, the counter is increased by 1. After the cell division, the ages of two new cells are set to 0. For an individual, when the bacterial age exceed the upper limit of its lifespan, it will be eliminated from the searching space. Meanwhile, the position of the dead cell will be tracked. From equation (12), it can be deduced that the bacteria tend to stay around the optima and the eliminated bacteria sink around the optima.

3.4 Simplified Quorum Sensing

The bacterial sensors are the receptor proteins that are signaled directly by external substances or via the piroplasmic substrate-binding proteins [10]. Bacteria use sensing to produce and secrete certain signaling compounds (called autoinducers or pheromone). These bacteria also have a receptor that can specifically detect the pheromone. When the inducer binds the receptor, it activates the transcription of certain genes, including those for inducer synthesis. Bacteria use signalling molecules which are released into the environment. As well as releasing the signalling molecules, bacteria are also able to measure the number of the molecules within a population.

Bacteria can obtain more nutrition around the optima. Based on this assumption, the density of the pheromone is increased at the position, where the fitness value is maximum. Each individual is attracted by the pheromone randomly. The mathematical description is:

$$\theta_i(t_c) = \theta_i(t_c) + \delta |g_{best}(t) - \theta_i(t_c)| \tag{22}$$

where δ indicates a gain parameter for the bacterial atteraction, $g_{best}(t)$ indicates the position of global best solution on tth optimization process, and $\theta_i(t_c)$ is the current position of the ith bacterium. Fig. 1 illustrates the structure of BFAVP.

4 Experiment Studies and Results

The BFAVP algorithm is tested on the standard IEEE 30-bus test system, which is adopted from [3]. The IEEE 30-bus case [3] represents a portion of the American Electric Power System (in the Midwestern US) as of December, 1961. There are 30 buses, 6 generators, and 40 branches in this model. To operate such a complex case, 23 variables are chosen to be the control variables, which is defined in section 2.

Table 1. Pseudo code for the BFAVP

Set $k = 0$;
Randomly initialize bacterial positions;
WHILE (termination conditions are not met)
FOR(each bacterium i)
Tumble! Generate a random tumbling angle Δ which is affected by quorum Sensing (equation (22)). Move to the new direction by equation (12), and draw energy $\Delta e_i(t_c)$ from the environment;
Run! For bacterium i, calculate the fitness value for the next step. If the current evaluation value is less than the value from last step, the bacteria will move towards an angle $\Delta(i)$ until it reach the the maximum step limit N_c. In each step, the energy $\Delta e_i(t_c)$ is absorbed from the environment;
END FOR
Reproduction: Select these bacteria which drew enough energy. Those bacteria divide into 2 individuals. The new cell takes half of the energy. The ages of the new cell is set to 0.
Elimination: These individuals which exceed the limit of lifespan will be eliminated, and their body positions are traced.
$k = k + 1$.
END WHILE

Fig. 2 illustrates the system layout of 30-bus. In this case, the objective function is the total fuel cost, which could be expressed as:

$$J = \sum_{i=1}^{N_G} f_i \tag{23}$$

$$f_i = a_i + b_i P_{G_i} + c_i P_{G_i}^2 \tag{24}$$

where f_i is the fuel cost (\$/h) of the ith generator, a_i, b_i and c_i are fuel cost coefficients, and P_{G_i} is the real power output generated by the ith generator.

The performance of BFAVP has been compared with the GA (Genetic Algorithm) [9] and PSO [1] in this section. A population of 50 individuals are used for GA and PSO. The initial population size of BFAVP is the same as PSO. However, during the evolutionary process, more individuals are reproduced in BFAVP. In order to prevent unnecessary increase of the population size, the maximal population size is set to 150 for BFAVP.

For PSO, the inertia weight ω is 0.73 and the acceleration factors c_1 and c_2 are both set to 2.05 which follows the recommendations from [12]. For BFAVP, the limit of a individual lifespan is 50 generations. The maximal length of a swim up along the gradient is 10 steps.

The best result, average results and standard deviation of GA, PSO and BFAVP from 30 runs are shown in Table 2. The standard deviation in table 2 are expressed as:

$$\sigma = \sqrt{\sum_{i=1}^{n} (g_{\mathrm{best_i}} - \frac{1}{n} \sum_{j=1}^{n} g_{\mathrm{best_j}})^2} \tag{25}$$

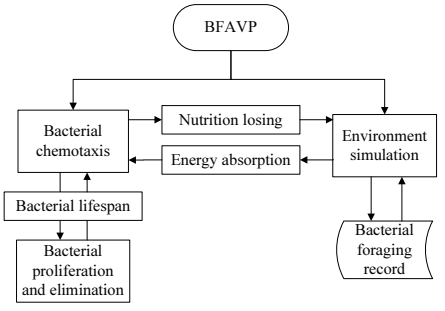


Fig. 1. BFAVP structure

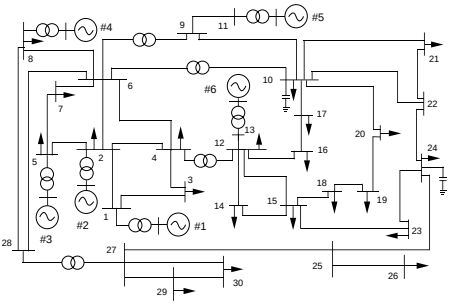


Fig. 2. IEEE 30-bus system

Table 2. Results from GA, PSO and BFAVP

	Best result	Average result	Standard deviation
GA	802.61	804.77	3.67
PSO	802.41	804.52	1.73
BFAVP	802.13	803.05	0.94

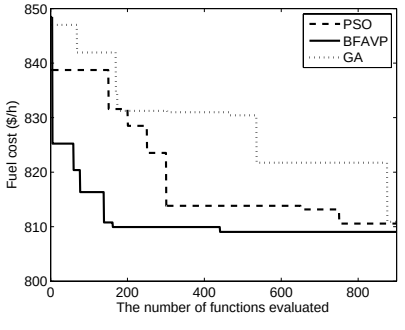


Fig. 3. Best solution for GA, PSO and BFAVP

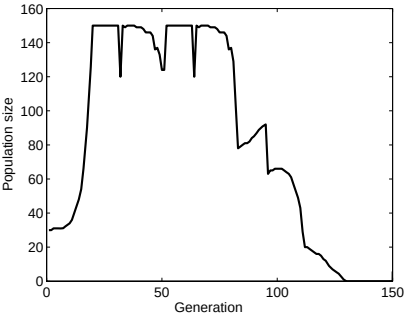


Fig. 4. Varying population size of BFAVP

where σ is the standard deviation of the best solution, g_{best_i} stands for the best solution of the ith experiment. In this case, there are 30 runs for each algorithm, so n equals to 30.

Fig. 3 illustrates the convergence process of GA, PSO, and BFAVP at the early stage. The horizontal axis stands for the number of functions evaluated during the optimization. The comparison is based on the same program running time in this figure. In this case, BFAVP converges faster than PSO and GA in the early period. PSO and GA are based on the stochastic searching, so the average fitness value is uncertain.

Fig. 4 is the growth curve of BFAVP during the OPF optimization. The horizontal axis is the number of generations during the program running. Lag

phase, logarithmic phase, and stationary phase can be discovered on this figure. In the early period, bacteria obtains enough energy for reproduction, which leads to an increasing population size. Then the population size remains stable until the nutrition has been consumed. At the last stage, population size is reduced due to the insufficient nutrition. The rugged deceasing population size in the deaths phase shows the bacteria converge to several optima.

5 Conclusion

A novel optimization algorithm, BFAVP, has been presented, which is based upon the bacterial foraging, proliferation, elimination, and quorum sensing behaviors. In contrast to the conventional EAs, BFAVP has a varying population which was introduced by simulating the phenomena of cells' split and contributes significantly to global search. The algorithm has also incorporated the tumble and run actions of chemotaxies processes, which greatly enhances the local search capability of the algorithm. The simulation studies have been carried out on an IEEE 30-bus system to tackle the OPF problem, which aims to minimize the fuel cost. The results show that BFAVP can provide improved solutions than PSO in terms of optimization accuracy and computation robustness.

References

1. Kennedy, J., and Eberhart, R. C., "Particle swarm optimization," *IEEE international Conference on Neural Networks,* **4**, (IEEE Press, 1995): 1942–1948.
2. Carpentier, J., "Contribution to the econimoc dispatch problem," **8**, (Bull. Soc. Franc. Elect, 1962): 431–447.
3. Alsac, O., and Scott, B., "Optimal load flow with steady state security," *IEEE Trans. on Power Appara. Syst.,* **PAS-93**, (1974): 745–751.
4. Bottero, M. H., Galiana, F. D., and Fahmideh-Vojdani A. R., "Economic dispatch using the reduced Hessian," *IEEE Trans. on Power Appara. Syst.,* **PAS-101**, (1982): 3679–3688.
5. Reid, G. F., and Hasdorf A. R., "Economic dispatch using quadratic programming," *IEEE Trans. on Power Appara. Syst.,* **PAS-92**, (1973): 2015-2023.
6. Stott, B., and Hobson E., "Power system security control calculation using linear programming," *IEEE Trans. on Power Appara. Syst.,* **PAS-97**, (1978): 1713-1731.
7. Momoh, J. A., and Zhu J. Z., "Improved interior point method for OPF problems," *IEEE Trans on Power Syst.,* **14**, (1999): 1114-1120.
8. Wei, H., Sasaki H., Kubokawa J., and Yokoyama R., "IAn interior point nonlinear programming for optimal power flow problems with a novel structure," *IEEE Trans on Power Syst.,* **13**, (1998): 870-877.
9. Bakirtzis, A. G., Biskas, P. N., Zoumas C. E., and Petridis, V.,"Optimal Power Flow by Enhanced Genetic Algorithm," *IEEE Transactions on Power Systems,* **17**(2), (May 2002): 229–236.
10. Passino, K. M., "Biomimicry of Bacterial Foraging for Distributed Optimization and Control," *IEEE Control Systems Magazine,* **22**(3), (June 2002): 52–67.

11. Bar Tana, J., Howlett B. J., and Koshland D. E., "Flagellar formation in *Escherichia coli* electron transport mutants," *Journal of Bacterial* **130**, (1977): 787–792.
12. Clerc, M., and Kennedy J., "The particle swarm: Explosion, stability, and convergence in a multi-dimensional complex space," *IEEE Trans. on Evolutionary Computation,* **6**, (2002): 58–73.

A Nomenclature

θ_{ij} voltage angle difference between bus i and j (rad)
B_{ij} transfer susceptance between bus i and j (p.u.)
G_{ij} transfer conductance between bus i and j (p.u.)
g_k conductance of branch k (p.u.)
N_0 the total number of total buses excluding slack bus
N_B the total number of total buses
N_C the total number of shunt compensators
N_D the total number of power demand buses
N_E the total number of network branches
N_G the total number of generator buses
N_i the total number of buses adjacent to bus i, including bus i
N_{PQ} the total number of PQ buses
N_{PV} the total number of PV buses
$N_Q^{\lim}$ the total number on buses on which injected reactive power outside limits
N_T the total number of transformer branches

$N_V^{\lim}$ the total number on buses on which voltages outside limits
P_{D_i} demanded active power at bus i (p.u.)
P_{G_i} injected active power at bus i (p.u,)
Q_{C_i} reactive power source installation at bus i (p.u.)
Q_{D_i} demanded reactive at bus i (p.u.)
Q_{G_i} injected reactive power at bus i (p.u.)
V_G voltage vector of PQ buses (p.u.)
T_i tap position at transformer i
V_i voltage magnitude at bus i (p.u.)
S_k apparent power flow in branch k (p.u.)

An Ant Algorithm for the Steiner Tree Problem in Graphs

Luc Luyet[1], Sacha Varone[2], and Nicolas Zufferey[3,⋆]

[1] Independent consultant, Geneva, Switzerland
[2] Haute École de Gestion de Genève, 1127 Carouge, Switzerland
[3] Faculté des Sciences de l'Administration,
Université Laval, Québec (QC), Canada, G1K 7P4
nicolas.zufferey@fsa.ulaval.ca

Abstract. The Steiner Tree Problem (STP) in graphs is a well-known NP-hard problem. It has regained attention due to the introduction of new telecommunication technologies, since it is the mathematical structure behind multi-cast communications. The goal of this paper is to design an ant algorithm (called ANT-STP) for the STP in graphs which is better than TM, which is a greedy constructive method for the STP proposed in [34]. We derive ANT-STP from TM as follows: each ant is a constructive heuristic close to TM, but the population of ants can collaborate by exchanging information by the use of the trail systems. In addition, the decision rule used by each individual ant is different from the decision rule used in TM. We compare TM and ANT-STP on a set of benchmark problems of the OR-Library.

1 Introduction

Let $G = (V, E, w)$ be a graph, V its vertex set, E its edge set, and w be a cost function which associates a positive cost $w[x, y]$ with each edge $[x, y] \in E$. We define a *tree* in a graph G as a subgraph s of G without any cycle which connects a subset of vertices of V. Given a subset R of V, the Steiner Tree Problem (STP) in graph G consists in finding a tree s in G which connects all the vertices in R (the mandatory vertices) at a minimum cost. Thus, the objective function f to minimize is simply $f(s) = \sum_{[x,y] \in s} w(x, y)$. The vertices in $V - R$ are called the *Steiner vertices*. The decision version of this problem has been shown in [22] to be NP-complete in the general case. As a result, the existing exact methods can not solve large instances and heuristic approaches are required to such instances likely to be encountered in real-life applications of the problem.

In telecommunication networks, data may have to be sent from one or more source(s), which are vertices in R and are called *terminals*, to multiple destinations, as in the case of conference calls or other applications sharing activities. The problem to send data to multiple destinations is known as the multicasting

⋆ Corresponding author.

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 42–51, 2007.
© Springer-Verlag Berlin Heidelberg 2007

routing problem [5], which has often additional delay constraints [27]. Online version of this problem, in which vertices can appear or disappear, can be found in [20], [29]. The weight function w is generally given by a multiple of a transmission capacity unit. Multicast routing problems occur in a variety of fields such as massive multi-player online role playing game [6], video or voice conferences [14], and collaborative virtual environments [13]. Several surveys describe data multicast techniques [25] or their associated combinatorial optimization methods [26].

STP in graphs also occurs in other fields, such as VLSI interconnect layout [4], [18]. VLSI designs generally consists in the interconnection of a set of pins, which must be made electrically connected by a set of wire segments. Although rectilinear metrics have to be considered in VLSI, there is a straightforward equivalence between STP with rectilinear metric and STP in graphs, since Hanan showed in [17] that only Steiner vertices formed by the intersections of vertical and horizontal lines through the vertices have to be considered.

Well known constructive heuristics for the STP in graphs are e.g. the ones proposed in [24], and in [34]. The latter starts with an empty solution s and successively connects a terminal vertex $x \notin s$ to s by the use of the shortest paths of value $SP(x, s)$ from x to s, where $SP(x, s) = \min_{y \in s} d(x, y)$, and where $d(x, y)$ is the length (or cost) of the path connecting x and y in G. At each step, the terminal vertex x such that $SP(x, s) = \min_{y \in R-s} SP(y, s)$ is added to the current partial solution s. Distributed heuristics have also been studied by several authors [23], [3], [32]. Among the most efficient algorithms, we can mention for example [28] and implementations of meta-heuristics such as genetic algorithms [21], [12], GRASP [31], tabu search [16], [30] or simulated annealing [11]. Heuristics for the STP are reviewed e.g. in [19], [36].

The solution space of the tabu search methods proposed in [16] consists in all the minimum spanning trees covering the mandatory vertices. Each element of such a solution space can thus be identified by its subset of Steiner vertices. In order to generate a neighbor solution s' from a solution s, the idea is to add in or remove from s a single Steiner vertex. The reverse of such a move is then tabu for some iterations. In the genetic algorithm presented in [12], the encoding is based on the use of the Distance Network Heuristic (DNH), the deterministic heuristic for the STP proposed in [24]. The genotype *gen* specifies a set of selected Steiner vertices, and a Steiner tree can be build from *gen* by applying DNH on the terminals and selected Steiner vertices.

In this paper, we present a distributed approach using an ant colony system to tackle the STP in graphs. We first describe a preprocessing step which hopefully will decrease the size of the solution space. We then present the main ideas of the ant algorithms and our ant heuristic for the STP. Finally, we show and discuss the obtained results on a set of benchmark instances.

2 Preprocessing Step

Often, the STP in graphs can be reduced using either exclusion or inclusion tests. The former identifies edges or non terminal vertices which do not belong

to at least one minimal Steiner tree, whereas the latter identifies edges or non terminal vertices which belong to all minimal Steiner trees. We choose some of the tests described in [19] for effectiveness reasons.

Four exclusion tests have been selected: non terminal of degree one, non terminal of degree two, long edge and paths with many terminals. The degree of a vertex x is the number of adjacent vertices to x. The Non Terminal of Degree One test (NTD1) removes vertices of degree one which are not in R, since minimal Steiner tree should not contain any non terminal vertices of degree one. The Non Terminal of Degree Two (NTD2) test is based on the following property: let v be a non terminal vertex of degree two, $[x, v]$ and $[v, y]$ be its two associated edges. Then

- if there is no edge $[x, y]$, then v and its two incident edges can be replaced by the edge $[x, y]$ with a weight $w[x, y] = w[x, v] + w[v, y]$;
- if there is an edge $[x, y]$ such that $w[x, y] \leq w[x, v] + w[v, y]$, then there exists a minimal Steiner tree which does not contain v; therefore v and its incident edges can be removed;
- if there is an edge $[x, y]$ such that $w[x, y] > w[x, v] + w[v, y]$, then no minimal Steiner tree contains $[x, y]$; therefore $[x, y]$ can be removed.

The Long Edge (LE) test removes each edge whose weight is strictly greater than the length of a shortest path between its extremities. The Path with many Terminals (PTm) test is a generalization of the LE test which involves the bottleneck Steiner distance. If all the vertices in a path, but its extremities, are non terminals, then this path is said to be elementary. Let P be a path between x and y. P is composed of one or more elementary paths. The longest elementary path in P is said to be the Steiner distance between x and y. The bottleneck Steiner distance $b(x, y)$ between x and y is the shortest Steiner distance among all the paths between x and y. The PTm test removes each edge whose weight is strictly greater than the bottleneck Steiner distance between its extremities.

Two inclusion tests have been selected: the Terminal of Degree One and the Nearest Vertex. The Terminal of Degree One (TD1) contracts each terminal vertex of degree one, since they belong to any (minimal) Steiner tree. The Nearest Vertex (NV) contracts the shortest edge $[z, x]$ incident to a terminal vertex z if the length of a second shortest edge $[z, y]$ is greater than or equal to $w[z, x] + SP[x, z']$, where z' is a terminal vertex.

The order in which those tests are performed is important since it has a strong impact on the computational time. Let SPM be a matrix, called the *shortest path matrix*, such that $SPM(x, y)$ equals the value of the shortest path between x and y. We propose the following preprocessing algorithm, which is close to the one proposed in [12].

While one of the following test is able to reduce G, do:
(a) perform TD1, NTD1, and NTD2;
(b) compute the shortest paths matrix SPM;
(c) perform PTm and LE;

Table 1. Reductions on B instances

	Initial Graph			Reduced Graph		
Instance I	$\|V\|$	$\|R\|$	$\|E\|$	$\|V\|$	$\|R\|$	$\|E\|$
B1	50	9	63	1	1	0
B2	50	13	63	7	4	11
B3	50	25	63	1	1	0
B4	50	9	100	25	6	42
B5	50	13	100	11	4	19
B6	50	25	100	20	9	36
B7	75	13	94	1	1	0
B8	75	19	94	1	1	0
B9	75	38	94	1	1	0
B10	75	13	150	44	9	92
B11	75	19	150	36	5	76
B12	75	38	150	18	9	33
B13	100	17	125	27	9	43
B14	100	25	125	21	8	37
B15	100	50	125	12	8	19
B16	100	17	200	60	9	135
B17	100	25	200	31	8	60
B18	100	50	200	15	7	23

Table 2. Reductions on C and D instances

	Initial graph	Gendreau *et al.*	Our reduced graph
I	$(\|R\|; \|E\|)$	$(\|V\|; \|R\|; \|E\|)$	$(\|V\|; \|R\|; \|E\|)$
C1	(5 ; 625)	(145 ; 5 ; 263)	(138 ; 5 ; 246)
C2	(10 ; 625)	(130 ; 8 ; 239)	(126 ; 8 ; 231)
C3	(83 ; 625)	(125 ;39 ;237)	(95 ;34 ; 178)
C4	(125 ; 625)	(116 ;42 ;233)	(74 ;29 ; 134)
C5	(250 ; 625)	(47 ;24 ;117)	(20 ;13 ; 36)
C6	(5 ; 1000)	(369 ;5 ;847)	(369 ;3 ;841)
C7	(10 ; 1000)	(382 ;9 ;869)	(380 ;9 ;857)
C8	(83 ; 1000)	(335 ;53 ;817)	(334 ;52 ;815)
C9	(125 ; 1000)	(351 ;80 ;834)	(322 ;75 ;711)
C11	(125 ; 2500)	(499 ;5 ;2184)	(498 ;5 ;2036)
C12*	(10 ; 2500)	(498 ;9 ;2236)	(498 ;9 ;2236)
C14	(5 ; 2500)	(414 ;68 ;1886)	(360 ;64 ;1000)
C16†	(5 ; 12,000)	(500 ;5 ;4740)	(500 ;5 ;4740)
C17*	(10 ; 12,000)	(500 ;10 ;4704)	(498 ;8 ;4685)
C18	(83 ; 12,000)	(483 ;67 ;4637)	(461 ;47 ;2575)
C19	(125 ; 12,000)	(433 ;58 ;3431)	(415 ;41 ;2016)
D1	(5 ; 1250)	(274 ;5 ;510)	(273 ;5 ;506)
D2	(10 ; 1250)	(285 ;10 ;523)	(283 ;10 ;519)
D3	(167 ; 1250)	(228 ;10 ;445)	(166 ;58 ;307)
D4	(250 ; 1250)	(180 ;81 ;376)	(105 ;47 ;196)
D12	(10 ; 1250)	(999 ;9 ;4669)	(994 ;9 ;3890)
D7	(10 ; 2000)	(754 ;10 ;1735)	(747 ;10 ;1709)
D8†	(167 ; 2000)	(732 ;124 ;1711)	(773 ;149 ;1753)
D9†	(250 ; 2000)	(660 ;157 ;1613)	(774 ;227 ;1749)
D16*	(5 ; 25,000)	(1000 ;5 ;10595)	(1000 ;5 ;10595)

We present the results of the above reduction procedure in Tables 1 and 2. We consider a set of the benchmark instances available from the OR-library [1], which can be found in [35]. Among the instances of type B, five have been solved to optimality using only the above reduction procedure, and the size of all instances but one has been reduced by half at least. All of them were preprocessed in less than two minutes on a computer *Silicon Graphics Indigo2 (195 MHz, IP28 processor)*. Instances of type C (characterized by $|V| = 500$) and D (characterized by $|V| = 1000$) have also been tested and compared favorably with those obtained in [16]. The drawback of our reduction procedure could rely on computational time, so that PTm may be inadequate for some instances. Therefore, on instances labeled by "*" in Table 2, the PTm test has been replaced by the LE test in order to avoid the computation of the bottleneck Steiner distance. In three cases labeled by "†", the procedure has been stopped after 1 hour of CPU time. In Table 2, we compare our reduction procedure with the one proposed in [16]. We can observe that we obtain better results except for instances D8 and D9.

3 Ant Algorithms

Evolutionary heuristics encompass various algorithms such as genetic algorithms, scatter search, ant systems and adaptive memory algorithms [2]. They can be defined as iterative procedures that use a central memory where information is collected during the search process. Ant colonies were first introduced in [9] and in [10]. In these methods, the central (long term) memory is modeled by a trail system. In the usual ant system, a population of ants is used, where each ant is a constructive heuristic able to build a solution step by step. At each step, an ant adds an element to the current partial solution. Each decision or *move* m is based on two ingredients: the *greedy force* $GF(m)$, which is the short term profit for the considered ant, and the *trails* $Tr(m)$, which represent the information obtained from other ants. Let M be the set of all the possible moves. The probability $p_k(m)$ that ant k chooses move m is given by

$$p_k(m) = \frac{GF(m)^\alpha \cdot Tr(m)^\beta}{\sum_{m' \in M_k(adm)} GF(m')^\alpha \cdot Tr(m')^\beta} \quad (1)$$

where α and β are parameters and $M_k(adm)$ is the set of admissible moves that ant k can perform at that time. In some ant algorithms [7], at each step and for a fixed value of parameter $p \in [0;1]$, a random number r is generated in $[0;1]$. If $r < p$, the chosen move is selected according to Equation (1), otherwise it is the one maximizing $p_k(m)$. When each ant of the population has built a solution, in which case we say that a generation has been performed, the trails are updated, for example as follows: $Tr(m) = \rho \cdot Tr(m) + (1-\rho) \cdot \Delta Tr(m)$, $\forall m \in M$, where $0 < \rho < 1$ is a parameter representing the evaporation of the trails, which is generally close or equal to 0.9, and $\Delta Tr(m)$ is a term which reinforces the trails left on move m by the ant population. That quantity is usually proportional to the number of times the ants performed move m, and to the quality of the obtained solutions when move m has been performed.

In some systems [7], the trails are updated more often (e.g. each time a single ant has built its solution) or are updated by only considering a subset of the ant population (e.g. the ants which provided the best solutions at the end of the current generation). In *hybrid ant systems*, the solutions provided by some ants may be improved using a local search technique. In the *max-min ant systems* [33], the authors proposed to normalize $GF(m)$ and $Tr(m)$ in order to better control these ingredients and thus the search process. An overview of ant algorithms can be found in [8].

4 Proposed Ant Algorithm

In order to propose an ant algorithm for the STP, we mainly have to define: a move, the greedy force of a move, the way to update the trails and the way to select a move.

Let N be the number of ants in the considered population. The role of a single ant k is to build a solution s_k step by step, starting with an empty solution s_k. At each step, as proposed in [34], we perform a move by connecting a terminal vertex $x \notin s_k$ to s_k using the shortest path between x and s_k. Thus, it is straightforward to define the greedy force of a terminal vertex $x \notin s_k$ as $GF(x) = \frac{1}{SP(x,s_k)}$.

To define $Tr(x)$, we associate a trail value $t(v)$ to each non terminal vertex v. $Tr(x)$ is then defined as the average value of t computed by considering each non terminal vertex v which belongs to the shortest path between x and s_k. We chose the average value of t instead of a summation in order to avoid to give too much importance to the terminal vertices which can be connected to s_k by a shortest path containing lots of non terminal vertices. At the end of each generation, the trails are *globally* updated as follows:

$$t(v) = (1 - \rho_g) \cdot t(v) + \rho_g \cdot \Delta t(v), \ \forall v \notin R,$$

where $\rho_g \in [0; 1]$ is a parameter. The reinforcement term $\Delta t(v)$ is updated as follows: $\Delta t(v) = \sum_k \Delta t_k(v)$, where the summation only holds for the N_{best} best ants of the current generation (i.e. the N_{best} ants with the smallest values of the objective function f), where N_{best} is a parameter. It is straightforward to set

$$\Delta t_k(v) = \begin{cases} \frac{1}{f(s_k)} & \text{if } v \in s_k; \\ 0 & \text{otherwise.} \end{cases}$$

Consequently, if $t(v)$ is large, it means on the one hand that most of the N_{best} best ants of the current generation have chosen v in their associated solutions, and on the other hand that these solutions have smaller values of f when compared to the other solutions built during the current generation.

In addition, assuming ant 1 to ant $k-1$ have respectively built their own solutions $s_1, \ldots, s_{k-1}$, when ant k has then built its solution s_k, the trails are *locally* updated as follows:

$$t(v) = \begin{cases} t(v) + \mu \cdot [1 - t(v)] \text{ if } f(s_k) \leq \min\limits_{i \in \{1,\ldots,k-1\}} f(s_i); \\ t(v) \text{ otherwise but } f(s_k) < \frac{1}{k-1} \sum\limits_{i=1}^{k-1} f(s_i); \\ \rho_l \cdot t(v) \text{ otherwise;} \end{cases}$$

where $\rho_l, \mu \in [0; 1]$ are parameters. We can observe that if s_k is the best solution of the current partial generation, the trails of non terminal vertices in s_k are reinforced. If it is not the case but s_k is better than the average quality of the solutions of the current partial generation, the trails of non terminal vertices in s_k keep the same value. Otherwise, such trails are reduced by an evaporation factor ρ_l.

At each step of the construction of a solution by a single ant, a random number r is generated in $[0; 1]$. If $r < p$ (where $p \in [0; 1]$ is a parameter), the chosen move is selected according to Equation (1) using "x" instead of "m" (where $x \in R - s_k$), otherwise (i.e. if $r \geq p$) it is the one maximizing $p_k(x)$. Note that in order to put some diversification in the general process, the first terminal vertex is always randomly selected when an ant starts to build its solution. In addition,

at the very beginning of the process, all the $t(v)$ values are initialized to 0.5. Thus, only the greedy force will guide the choices of the first ant in the first generation. Equivalently, such an ant simply applies the method proposed in [34].

We have now all the ingredients which are necessary to design our general heuristic for the STP in graphs.

1. Initialize $t(v) = 0.5$ for all $v \notin R$;
2. Initialize the parameters: $N, N_{best}, \rho_g, \rho_l, \mu, p, \alpha, \beta$;
3. Set $f^* = \infty$;
4. While 500 generations without improvement of f^* have not been performed, do:
   (a) for $k = 1$ to N, do:
      i. initialize s_k with a randomly chosen terminal z;
      ii. successively connect a terminal vertex x to s_k by the use of the shortest path from x to s_k; repeat this step until all terminal vertices are in s_k;
      iii. update the trails (locally);
      iv. if $f(s_k) < f^*$, set $f^* = f(s_k)$ and $s^* = s_k$;
   (b) update the trails (globally);
5. Return solution s^* of value f^*;

5 Numerical Results

It is important to mention that all the ingredients introduced in the previous section (namely $GF(x), Tr(x), t(v), \Delta t(v)$) are always normalized in interval $[0; 1]$. Such normalization leads to a better control on the search. In addition, preliminary experiments showed that the following parameter setting is appropriate: $N = 30, N_{best} = 3$ (thus only the best 10% of the ants are involved to update the trails), $\rho_g = \rho_l = 0.9$ as it is the case in most of the ant algorithms, $\mu = 0.1, p = 0.5, \alpha = 1$ and $\beta = 0.02$. Therefore, it is better to give more weight to the greedy force rather than to the trails. The stopping condition of our algorithm is a maximum number of generations without improvement of the best solution found so far during the search. Such a number is fixed to 500 and all the experiments were performed on a *Silicon Graphics Indigo2 (195 MHz, IP28 processor)*.

We always start our general heuristic by performing the reductions proposed above. For the B instances presented in Table 1, we always got the optimal solution in less than one second. Thus, we will not focus anymore on these instances. Such observation does not hold for the instances presented in Table 2. For these instances, the results are given in Table 3, in which we provide the optimal value of f for each instance (this is denoted by OPT). In this Table, we compare three methods: TM, which is a multi-start constructive method proposed in [34]; TabuGLS, which is an efficient tabu search method proposed in [16]; and ANT-STP, which is our ant algorithm.

We observed that instances C1, C2, C3, C5, C6, C7, C11, C12, C16, C17, D1, D7, D11, D12, and D16 were optimally solved in a few seconds by TM, TabuGLS and ANT-STP. Thus, we do not provide any information on such instances in

Table 3. For each method and each remaining instance I, we only indicate a result if the considered method is not able to find the optimal solution.

On the one hand, it is interesting to compare our method with TM because if we ignore the trails in ANT-STP, we obtain a method which is close to TM. In Table 3, we actually provide the best obtained results if we perform TM $|V|$ times. Each time, we restart the method by initializing the current partial solution with a different vertex in V. In other words, if we allow the same amount of CPU time to TM and ANT-STP, we show that all the ingredient we add to TM in order to derive ANT-STP are useful. Consequently, the collaboration between the ants is well defined.

On the other hand, as the methods TabuGLS and ANT-STP start to work on their associated reduced instances, we will be able to measure the effectiveness of our method when compared to one of the best state-of-the-art method. In the two last columns, we respectively give the CPU times (in seconds and ignoring the time spent for the reductions) T-TabuGLS and T-ANT-STP associated with TabuGLS and ANT-STP. Note that we only indicate such times if they are larger than 60 seconds, and that the TabuGLS method was performed on a computer *Ultra Sparc 1 (170)*, which is comparable to the computer we used for our experiments.

Table 3. Obtained results

I	OPT	TM	TabuGLS	ANT-STP	T-TabuGLS	T-ANT-STP
C4	1079	1080				
C8	509	510			23	74
C9	707	714	708	708	50	3189
C14	323	326	324		37	150
C18	113	122		116	53	943
C19	146	153		169	39	463
D2	220	221			7	
D3	1565	1570	1567	1565	16	266
D4	1935	1936				
D8	1072	1088	1076	1084	244	2312
D9	1448	1471	1451	1465	331	1929

We can first observe that ANT-STP always obtained better results than TM (except on instance C19). If we compare ANT-STP with TabuGLS, we can remark that ANT-STP performed better on instances C14 and D3, but TabuGLS was better on instances C18, C19, C8, and D9. On every other instance, both methods got similar results. However, TabuGLS generally consumes less CPU time than ANT-STP.

6 Conclusion

In this paper, we have presented an ant algorithm, called ANT-STP, to solve the Steiner tree problem in graphs. The role of each ant is to build a solution step by step as in the constructive method TM proposed in [34]. If we ignore the trails, ANT-STP and TM could be considered as similar methods, because at each step of the construction, a terminal vertex x is added to the current partial solution s by the use of the shortest path between x and s. Because ANT-STP was very favorably compared to TM, it is obvious that all the ingredients we

add in TM in order to design ANT-STP are useful. These ingredients mainly are the short term and long term memory, i.e. the trail systems (remember that trails are updated locally and globally), and the way to select the next terminal to add in the current partial solution.

To assess the efficiency of the procedure, ANT-STP was applied to a set of benchmark instances for which the optimal solution is known, and the results were compared to TabuGLS, which is a tabu search heuristic proposed in [16]. We observed that ANT-STP and TabuGLS obtained comparable results but ANT-STP needs more time to get such results.

Finally, we would like to mention that, to improve overall system efficiency, ant algorithms are often enriched with extra capabilities such as lookahead, local optimization, backtracking, and so on (which cannot be found in real ants). This is the case in many implementations, where constructive ant systems have been hybridized with local optimization procedures (see, e.g., [7], [15], [33]]). In contrast with such hybrid ant heuristics, ANT-STP is competitive without using any local search procedure to improve the solutions built by the ants.

References

1. Beasley, J.: Or-library: Distributing test problems by electronic mail. Journal of the Operational Research Society **41** (1990) 1069–1072
2. Calegari, P., Coray, C., Hertz, A., Kobler, D., and Kuonen, P.: A taxonomy of evolutionary algorithms in combinatorial optimization. Journal of Heuristics **5** (1999) 145–158
3. Chen, G., Houle, M., and Kuo, M.: The steiner problem in distributed computing systems. Information Sciences **74(1)** (1993) 73–96.
4. Cong, J., He, L., Koh, C., and Madden, P.: Performance optimization of vlsi interconnect layout. Integration: the VLSI Journal **21** (1996) 1–94
5. Deering, S., and Cheriton, D.: Multicast routing in datagram internetworks and extended lans. ACM Transaction on Computer Systems **8(2)** (1990) 85–110
6. Diot, C., and Gautier, L.: A distributed architecture for multiplayer interactive applications on the internet. IEEE Network **13(4)** (1999) 6–15
7. Dorigo, M., and Gambardella, L.: Ant colony system: a cooperative learning approach to the traveling salesman problem. IEEE Transactions on Evolutionary Computation **1** (1997) 53–66
8. Dorigo, M., Di Caro, G., and Gambardella, L.: Ant algorithms for discrete optimization. Artificial Life **5** (1999) 137–172
9. Dorigo, M., Maniezzo, V., and Colorni, A.: Positive feedback as a search strategy. Technical Report 91-016, Politecnico di Milano, Dipartimento di Elettronica, Italy (1991)
10. Dorigo, M.: Optimization, learning and natural algorithms (in Italian). Ph.D. Dissertation, Politecnico di Milano, Dipartimento di Elettronica, Italy (1992)
11. Dowsland, K.: Hill-climbing simulated annealing and the steiner problem in graphs. Engineering Optimization **17** (1991) 91–107
12. Esbensen, H.: Computing near-optimal solutions to the steiner problem in a graph using a genetic algorithm. Networks: An International Journal **26** (1995) 173–185
13. Fisher, H.: Multicast issues for collaborative virtual environments. IEEE Computer Graphics and Applications **22(5)** (2002) 68–75

14. Foreman, D.: Managing data in distributed multimedia conferencing applications. IEEE Multimedia 9(4) (2002) 30–37
15. Gambardella, L. M., and Dorigo, M.: HAS-SOP: An hybrid ant system for the sequential ordering problem. Tech. Rep. 11-97, Lugano, Switzerland: IDSIA (1997)
16. Gendreau, M., Larochelle, J.-F., and Sansò, B.: A tabu search heuristic for the steiner tree problem. Networks **34(2)** (1999) 162–172
17. Hanan, M.: On steiner's problem with rectilinear distance. SIAM Journal of Applied Mathematics **14(2)** (1966) 255–265
18. Hu, J., and Sapatnekar, S. S.: A survey on multi-net global routing for integrated circuits. Integration: The VLSI Journal **31** (2001) 1–49
19. Hwang, F., Richards, D., and Winter, R.: The Steiner tree problem, volume 53 of Ann. of Discrete Math. Amsterdam: North-Holland (1992)
20. Imase, M., and Waxman, B. M.: Dynamic steiner tree problem. SIAM J. Discrete Math. **4(3)** (1991) 369–384
21. Kapsalis, A., Rayward-Smith, V., and Smith, G.: Solving the graphical steiner tree problem using genetic algorithms. Journal of the Operational Research Society **44** (1993) 397–406
22. Karp, R. M.: Reducibility among combinatorial problems. Plenum Press. (1972) 85–103
23. Kompella, V., Pasquale, J., and Polyzos, G.: Two distributed algorithms for the constrained steiner tree problem. In Proceedings of the Second International Conference on Computer Communications and Networking (ICCCN'93) (1993) 343–349
24. Kou, L., Markowsky, G., and Berman, L.: A fast algorithm for steiner trees. Acta Informatica **15** (1981) 141–145
25. Mir, N.: A survey of data multicast techniques, architectures, and algorithms. IEEE Communications Magazine **39(9)** (2001) 164–170
26. Oliveira, C. A. S., and Pardalos, P. M.: A survey of combinatorial optimization problems in multicast routing. Comput. Oper. Res. **32(8)** (2005) 1953–1981
27. Pasquale, J., Polyzos, G. C., and Xylomenos, G.: The multimedia multicasting problem. Multimedia Systems **6(1)** (1998) 43–59
28. Polzin, T., and Daneshmand, S. V.: The multimedia multicasting problem. Multimedia Systems **6(1)** (1998) 43–59
29. Raghavan, S., Manimaran, G., and Murthy, C. S. R.: A rearrangeable algorithm for the construction delay-constrained dynamic multicast trees. IEEE/ACM Trans. Netw. **7(4)** (1999) 514–529
30. Ribeiro, C. C., and Souza, M. C. D.: Tabu search for the steiner problem in graphs. Networks **36(2)** (2000) 138–146
31. Ribeiro, C. C., Uchoa, E., and Werneck, R. F.: A hybrid grasp with perturbations for the steiner problem in graphs. INFORMS J. on Computing **14(3)** (2002) 228–246
32. Shaikh, A., and Shin, K. G.: Destination-driven routing for low-cost multicast. IEEE Journal of Selected Areas in Communications **15(3)** (1997) 373–381
33. Stuetzle, T., and Hoos, H.: The MAX-MIN Ant System and local search for the traveling salesman problem. In T. Baeck, Z. Michalewicz, and X. Yao (Eds.) Proceedings of IEEE-ICEC-EPS97. Piscataway, NJ: IEEE Press (1997) 309-314
34. Takahashi, H., and Matsuyama, A.: An approximate solution for the steiner problem in graphs. Math. Japonica **24(6)** (1980) 573–577
35. Voss, S., Martin, A., and Koch, T.: Steinlib testdata library. online
36. Voss, S.: Steiner's problem in graphs: Heuristic methods. Discrete Applied Mathematics **40** (1992) 45–72

Message Authentication Protocol Based on Cellular Automata

A. Martín del Rey

Department of Applied Mathematics, E.P.S. de Ávila
Universidad de Salamanca
C/Hornos Caleros 50, 05003-Ávila, Spain
delrey@usal.es

Abstract. The main goal of this work is to study the design of message authentication protocols by using cellular automata. Specifically, memory cellular automata with linear transition functions are considered. It is shown that the proposed protocol is secure against different cryptanalytic attacks.

1 Introduction

Data integrity and message authentication are fundamental goals of cryptography. In this sense, hash functions play an important role to meet these objectives. Roughly speaking, hash functions map m-bitlength messages into k-bitlength messages ($k << m$), called hashes, such that it is computationally infeasible to find two original messages with the same hash.

In the last years several hash functions have been proposed in the literature ([10]): MD4, MD5, SHA-1, etc. These functions have only one input: the message M, and only one output: the hash R corresponding to M. Nevertheless, there exists another type of hash functions which possess two inputs: the original message M and a secret key K, and only one output: the hash R. They are called message authentication codes (MAC). Consequently, in this case, only the users with the correct key can compute the hash.

For example, message authentication protocols can be used to check if the files stored into a hard drive have been modified in some way: The authorized user computes the file's hash using the secret key. Subsequently, he compares the hash obtained with the original one (which are securely stored). If they are equal then no modification has been done, otherwise the file has been changed. There are several ways to design MAC: some of them are suitable modifications of hash functions with the addition of a secret key; others are based on symmetric cryptosystems: stream ciphers or block ciphers (see [10]), etc.

The main goal of this work is to study the use of cellular automata in order to design message authentication protocols. Cellular automata (CA for short) were originally conceived by Ulam and von Neumann in the 1940s to provide a formal framework for investigating the behavior of complex, extended systems, that is, its main goal was to design self-replicant artificial systems that are also

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 52–60, 2007.
© Springer-Verlag Berlin Heidelberg 2007

computationally universal ([16]). Cellular automata are dynamical systems in which space and time are discrete. A CA consists of an array of cells, each of which can be in one of a finite number of possible states, updated synchronously in discrete time steps, according to a local and identical interaction rule. The state of a cell at the next time step is determined by the current states of a surrounding neighborhood of cells (see, for example, [15,18]).

CA possess several fundamental properties of the physical world: they are massively parallel, homogenenous and all interactions are local. Other physical properties such as reversibility and conservation laws can be programmed by choosing the local updated rule properly. It is not surprising that nature systems have been successfully simulating using CA models. In this sense, there are several applications in biological systems (model of competitive growth of species, study of morphogenesis in simple cellular systems, modelization of a tumor growth, etc.), in economic systems, in environmental and ecological systems (design of forest fire and epidemic spreading models, study of vegetation dynamics behaviour, study of soil erosion by water, etc.), in engineering and complex industrial systems (traffic system study, image processing, machine learning and control, coding theory, cryptography,...), etc.

The design of cryptographic protocols by means of CA goes back to middles 80's when S. Wolfram proposed the elementary cellular automata with rule number 30 as pseudorandom bit generator (see [17]). Since then, a lot of CA-based cryptographic algorithms have been appeared: secret key cryptosystems (see, for example [2,7,13,14]), public key cryptosystems ([8]), secret sharing schemes ([1,9]), design of hash functions ([5,6,11]) and MAC ([12]).

The MAC protocol proposed in [12] involves additive cellular automata and their duals; specifically single attractor cellular automata (SACA for short) and their duals are considered. Those cellular automata are not reversible and consequently, different initial configurations yield the same final configuration. That is a non-desirable property when MAC protocols are designed because many initial messages can produce the same hash. In this work we propose the use of simplest CA: the CA with rule number 150. It is a reversible cellular automata whose local transition function is very simple. It is shown that our scheme is robust against the more important cryptanalytic attacks, and experimental results establish the higher speed of execution of our scheme. Moreover, it is shown that it is not necessary the presence of their duals to obtain a robust and efficient protocol.

The rest of the work is organized as follows: In section 2 the basic theory about cellular automata is introduced; the message authentication protocol is shown in section 3 with some examples and its computational complexity; the security analysis is presented in section 4, and finally the conclusions are shown in section 5.

2 Cellular Automata

Cellular automata are simple models of computation capable to simulate complex physical, biological and environmental phenomena (see, for example [18]).

Specifically, a CA is a finite state machine formed by a collection of k memory units called cells which are endowes with a state at each time step. In this work, the state set is given by the finite field $GF(2^n)$ where $n \geq 1$ and these states change according to a deterministic function. So, if s_i^t stands for the state of the i-th cell at time step t, then

$$s_i^{t+1} = f\left(s_{i-1}^t, s_i^t, s_{i+1}^t\right) \in GF(2^n), \quad 1 \leq i \leq k. \tag{1}$$

That is, the state of each cell at a particular time step t depends on the states of the own cell and its two neighbor cells at time $t-1$. In this sense, it is said that the neighbor of each cell is formed by the own cell and its two adjacent cells.

Moreover, to assure a well-defined evolution of the CA, it is necessary to establish null boundary conditions:

$$s_1^{t+1} = f\left(0, s_1^t, s_2^t\right), \quad s_k^{t+1} = f\left(s_{k-1}^t, s_k^t, 0\right). \tag{2}$$

A CA is said to be linear when its local transition function is of the following form:

$$s_i^{t+1} = \alpha s_{i-1}^t + \beta s_i^t + \gamma s_{i+1}^t \,(\mathrm{mod}\, 2^n), \tag{3}$$

with $\alpha, \beta, \gamma \in GF(2^n)$.

The configuration of the CA at time step t is the vector

$$C^t \in GF(2^n) \times \overset{(k)}{\cdots} \times GF(2^n), \tag{4}$$

which is formed by all the states of the cells of the CA at time t.

The evolution of a linear CA can be interpreted in terms of linear algebra:

$$C^{t+1} = T \cdot C^t \pmod{2^n}, \tag{5}$$

where T the characteristic matrix of the CA, which is the following tridiagonal matrix:

$$T = \begin{pmatrix} \beta & \gamma & 0 & \cdots & \cdots & 0 \\ \alpha & \beta & \gamma & \ddots & & \vdots \\ 0 & \alpha & \beta & \gamma & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & & \ddots & \alpha & \beta & \gamma \\ 0 & \cdots & \cdots & 0 & \alpha & \beta \end{pmatrix}. \tag{6}$$

When $\alpha, \beta, \gamma = 1$, the CA obtained is called elementary CA with rule number 150 whose applications to Cryptography and Coding Theory are very important (see, for example, [4]).

Note that, in general, the evolution of a CA considers that the configuration at time $t+1$ of the CA depends only on its configuration at the previous time step t, that is,

$$C^{t+1} = \Phi\left(C^t\right), \tag{7}$$

where Φ is the global transition function of the CA. This is the standard paradigm for the evolution of CA; nevertheless, one can also assume that C^{t+1} not only depends on C^t, but also on the configurations at the m previous time steps:

$$C^{t+1} = \Psi\left(C^t, C^{t-1}, \ldots, C^{t-m+1}\right). \tag{8}$$

This new kind of CA is called m-th order memory CA (m-MCA for short). Specifically, this work deals with 2-MCA of the form:

$$C^{t+1} = \Phi_0\left(C^t\right) + \Phi_1\left(C^{t-1}\right), \tag{9}$$

and with 3-MCA such that:

$$C^{t+1} = \Phi_0\left(C^t\right) + \Phi_1\left(C^{t-1}\right) + \Phi_2\left(C^{t-2}\right), \tag{10}$$

where Φ_0, Φ_1, Φ_2 stand for global transition functions defined by local linear transition functions. That is, in the first case:

$$C^{t+1} = T_0 \cdot C^t + T_1 \cdot C^{t-1} \pmod{2^n} \tag{11}$$

and in the second case:

$$C^{t+1} = T_0 \cdot C^t + T_1 \cdot C^{t-1} + T_2 \cdot C^{t-2} \pmod{2^n}, \tag{12}$$

where

$$T_i = \begin{pmatrix} \beta_i & \gamma_i & 0 & \cdots & \cdots & 0 \\ \alpha_i & \beta_i & \gamma_i & \ddots & & \vdots \\ 0 & \alpha_i & \beta_i & \gamma_i & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & & \ddots & \alpha_i & \beta_i & \gamma_i \\ 0 & \cdots & \cdots & 0 & \alpha_i & \beta_i \end{pmatrix}, \; 0 \leq i \leq 2, \tag{13}$$

where $\alpha_i, \beta_i, \gamma_i \in GF\left(2^n\right)$.

3 The Message Authentication Protocol

3.1 The Algorithm

The proposed protocol to authenticate digital messages is formed by four stages:

- Initialization phase.
- Preprocessing phase.
- Processing phase.
- Final phase.

Specifically, each phase is as follows:

Initialization phase. In this phase the secret key K and the three cellular automata to be used in the protocol are chosen.

The state set is $GF(2^n)$ where $n = 1, 2, 4, 8, 16$ ó 32, and the local transition functions of the CA are the following:

1. *First CA*, Γ: It is a 2-MCA defined by (9) with $\alpha_i = \beta_i = \gamma_i = 1$, for every i, that is, their functions Φ_0, Φ_1 are the global functions of an elementary CA with rule number 150.
2. *Second CA*, Σ: It is a 3-MCA defined by (10) with $\alpha_i = \beta_i = \gamma_i = 1$ for every i. As in the previous case , the functions Φ_0, Φ_1, Φ_2, are the global functions associated to CA with rule number 150.
3. *Third CA*, Ω: It is the elementary CA defined by the rule number 150.

Moreover, a secret key of 160-bitlength must be chosen.

Preprocessing phase. The second phase of the algorithm is given by the following steps:

1. let $M = (m_1, \ldots, m_k)$ be the message to be hashed such that $m_i \in GF(2^n)$, $1 \leq i \leq k$.
2. Append to M the element $m_{k+1} = 2^n - 1 \in GF(2^n)$ and as few random elements of $GF(2^n)$ as necessary to obtain a string, $\tilde{M}$, whose bitlength is a multiple of 160.
3. Partition the obtained string into p blocks of 160-bitlength:

$$\tilde{M} = M_1 \,||\, M_2 \,||\, \ldots \,||\, M_p. \tag{14}$$

Processing phase. In the third phase of the algorithm, the hash of the original message is computed as follows:

1. Compute $H_1 = \Gamma(M_1, K)$.
2. Compute $H_i = \Sigma(M_i, H_{i-1}, K)$ with $2 \leq i \leq p$.

Final phase. To finish the algorithm, the following computation is done:

1. Compute $R = \Omega^{160/n}(H_p)$, that is, the elementary CA with rule number 150 is applyied $160/n$ times in order to obtain the sufficient diffusion level.

3.2 Examples

For the sake of simplicity, let us consider the 256-bitlength message, M, given by:

$$\texttt{ef3926978551bc662984218a69a7daef4a49652b33f1f3f83c5c8c48786e0d18f}, \tag{15}$$

and set

$$K = \texttt{d4a6edb72d0d9e3c1fec32c1cd6e62e94da1f509}, \tag{16}$$

the secret 160-bitlength key. The hashes obtained for different state sets are shown in Table 1:

Table 1. Different hashes obtained from the same message

$GF(2^n)$	Hash
$n = 1$	5190ab6b0bf51675b413041c18bbcf97e38125a9
$n = 2$	19a13d06d126b3d495fc1a8785aa5617044f0772
$n = 4$	45903b64d17bb5671280927c39859568006d6757
$n = 8$	7918fc27788a354efaa8edade9aebb3878bc46f3
$n = 16$	d66e2e5f2aebd83d775de831eaf67b43679d816b

3.3 Running Times

The algorithm stated in the last section has been computationally implemented using the computer algebra system Mathematica^{TM} (release 5.0). The CPU computation times (in seconds) obtained for different message lengths (L) and different state sets are shown in Table 2. The computer used in this test is 1.87 GHz Pentium M with 2GB RAM memory.

Table 2. Computation times

$GF(2^n)$	$L = 256$	$L = 512$	$L = 1024$	$L = 10^4$	$L = 10^5$	$L = 10^6$
$n = 4$	0	0	0.016	0.062	0.328	2.234
$n = 8$	0	0	0	0.015	0.265	1.172
$n = 16$	0	0	0	0	0.078	0.75

Moreover, comparative results for CA-based authentication algoritm, CAA, (see [12]), MD5 hash function and our proposed protocol in the case $n = 4$ are displayed in Table 3.

Table 3. Computation times of CAA protocol, MD5 hash function and our protocol

$GF(2^n)$	$L = 1024$	$L = 10^4$
CAA	0.040	0.105
$MD5$	0.0549	0.165
Our protocol	0.016	0.062

These experimental results establish the higher speed of execution of our scheme.

4 Security Analysis

There are two types of brutte-force attacks on hash functions: (1) Given a hash obtained from a message M, the opponent try to find a different message M' with the same size; (2) The cryptanalyst try to find two messages with the same

hash (birthday attack). In our case as the bitlength of the key is of 160 bits, it is necessary to analyze 2^{160} messages to find another with the same hash; moreover it is necessary to check 2^{80} messages to find a pair with the same hash. As a consequence, the proposed protocol is secure against these two attacks.

Another cryptanalytic attacks take advantage of some weakness of the protocol about the sensitivity to initial conditions: The message and the keys. In this sense it is very convenient that slightly different initial messages (for example, only one-bit difference) yield very different hashes (for example, the Hamming distance between them must be of 80 bits if the the hash is 160-bitlength). In Table 4 the results are shown when we compute the hashes of 100 pairs of 10.000-bitlength messages with one-bit difference (Test I), and when we compute the hashes of 100 different messages with one-bit difference keys (test II).

Table 4. Sensitivity to initial conditions

	Test I	Test II
$GF\,(2^n)$	Hamming distance	Hamming distance
$n = 1$	5	38
$n = 2$	15	68
$n = 4$	69	72
$n = 8$	73	80
$n = 16$	84	81

Note that the best results are obtained for those CA with state sets given by $n = 8$ and $n = 16$. That is why as greater is n as stronger is the connection between the cells, which favours the diffusion process. In this sense, a CA with k cells and defined by the state set $GF\,(2^n)$ is equivalent to a CA with $k \times n$ cells defined over $GF\,(2)$ with neighborhoods of $3 \times n$ cells (see [4]).

Finally, we will study the robustness of the proposed protocol against differential cryptanalysis. This attack analyzes the plaintext pairs along with their corresponding hashes pairs to identify the correlations that would enable identification of the secret key used. For example, let us suppose that the bitlength of the original message, M, is of 160 bits and set M', another message which differs from M in $d\,(M, M') = 4$ bits:

$$M = \texttt{3e442bb4199d5f068ea3c7453030ddf9530d7c93}, \tag{17}$$

$$M' = \texttt{3e442bb4189d5f068ea2c7453030dcf9530d7c92}. \tag{18}$$

Their respective hashes are the following:

$$R = \texttt{7f7105bcdbd33fb1917738ae5ccde1e1e8e4163a}, \tag{19}$$

$$R' = \texttt{f9c427110929ff1dd088733f0f6bfcd3f7b484cf}, \tag{20}$$

which differs in $d\,(R, R') = 79$ bits.

Then, for every pair of messages, M and M', such that $d(M, M') = 4$, we compute the values $d(R, R')$ and taking into account their distribution, we compute the standard deviation σ. If $\sigma < 10\%$, then the protocol is secure against differential cryptanalysis (see [3]). In Table 5 the results obtained using the CA defined over $GF(2^{16})$ are shown. Note that, in this case, the protocol is secure.

Table 5. Differential cryptanalysis

$d(M, M')$	8	16	20	24	30	32	40	48
σ	8.88	7.45	7.89	6.84	6.36	6.68	6.35	5.77

5 Conclusions

In this work the use of cellular automata in order to design message authentication protocols has been studied. Specifically, it deals with memory cellular automata whose local transition functions are linear defined over the state set $GF(2^n)$, where $n = 1, 2, 4, 8, 16, 32$.

The secret key is 160-bitlength and the protocol involves three cellular automata: the elementary cellular automata with rule number 150, and two memory cellular automata based on the first one. The bitlength of the hashes computed is 160.

It is shown that the proposed protocol is secure against brutte-force attacks, differential cryptanalysis and sensitive tests.

Acknowledgements

This work has been partially supported by the Consejería de Educación (Junta de Castilla y León, Spain) under grant SA110A06, and by Ministerio de Educación y Ciencia (Spain) under grant SEG2004-02418.

References

1. Álvarez, G., Hernández, A., Hernández, L., Martín, A.: A secure scheme to share secret color images, Comput. Phys. Comm. **173** (2005) 9–16.
2. Bao, F.: Cryptanalysis of Partially Known Cellular Automaton Cryptosystem, IEEE Trans. Comput. **53** (2004) 1493–1497.
3. Biham, E., Shamir, A.: Differential cryptanalysis of DES-like cryptosystems, J. Cryptology **4** (1991) 3–72.
4. Chaudhuri, P., Chowdhury, D., Nandi, S., Chattopadhyay, S.: Additive cellular automata. Theory and Applications. Volume 1, IEEE Computer Society Press, Los Alamitos, 1997.
5. Damgard, I.: A design principle for hash functions, Advances in Cryptology: Proc. of Crypto'89, LNCS **435** (1990) 416–427.

6. Dasgupta, P., Chattopadhyay, S., Sengupta, I.: Theory and application of non-group cellular automata for message authentication, J. Syst. Architecture **47** (2001) 383–404.
7. Fúster-Sabater, A., de la Guía-Martínez, D.: Cellular automata applications to the linearization of stream cipher generators, Proc. of ACRI 2004, LNCS **3305** (2004) 612–621.
8. Guan, P.: Cellular automaton public-key cryptosystem, Complex Systems, **1** (1987) 51–57.
9. Martín, A., Pereira , J., Rodríguez, G.: A secret sharing scheme based on cellular automata, Appl. Math. Comput. **170** (2005) 1356–1364.
10. Menezes, A., van Oorschot, P., Vanstone, S.: Handbook of Applied Cryptography, CRC Press, Boca Raton, FL, 1997.
11. Mihaljevic, M., Zheng, Y., Imai, H.: A family of fast dedicated one-way hash functions based on linear cellular automata over $GF(q)$, IEICE Trans. Fundamentals **E82-A** (1999) 40–47.
12. Mukherjee, M., Ganguly, N., Chaudhuri, P.P.: Cellular automata based authentication, Proc. of ACRI 2002, LNCS **2493** (2002) 259–269.
13. Nandi, S., Kar, B.K., Chaudhuri, P.P.: Theory and applications of cellular automata in cryptography, IEEE Trans. Comput. **43** (1994) 1346–1357.
14. Seredynski, M., Bouvry, P.: Block encryption using reversible cellular automata, Proc. of ACRI 2004, LNCS **3305** (2004) 785–792.
15. Toffoli, T., Margolus, N.: Cellular automata machines, The MIT Press, Cambridge, MA, 1987.
16. von Neumann, J.: Theory of self-reproducing automata, (edited and completed by A.W. Burks) University of Illinois Press, Illinois, 1966.
17. Wolfram, S.: Cryptography with cellular automata, Advances in Cryptology: Proc. of Crypto'85, LNCS **218** (1986) 429-432.
18. Wolfram, S.: A new kind of science, Wolfram Media, Champaign, Illinois, 2002.

An Adaptive Global-Local Memetic Algorithm to Discover Resources in P2P Networks

Ferrante Neri[1,2], Niko Kotilainen[1], and Mikko Vapa[1]

[1] Department of Mathematical Information Technology, Agora, University of Jyväskylä, FI-40014, Finland
{neferran,npkotila,mikvapa}@jyu.fi
[2] Dipartimento di Elettrotecnica ed Elettronica, Politecnico di Bari, Via E. Orabona 4, 70125, Italy
neri@deemail.poliba.it

Abstract. This paper proposes a neural network based approach for solving the resource discovery problem in Peer to Peer (P2P) networks and an Adaptive Global Local Memetic Algorithm (AGLMA) for performing the training of the neural network. This training is very challenging due to the large number of weights and noise caused by the dynamic neural network testing. The AGLMA is a memetic algorithm consisting of an evolutionary framework which adaptively employs two local searchers having different exploration logic and pivot rules. Furthermore, the AGLMA makes an adaptive noise compensation by means of explicit averaging on the fitness values and a dynamic population sizing which aims to follow the necessity of the optimization process. The numerical results demonstrate that the proposed computational intelligence approach leads to an efficient resource discovery strategy and that the AGLMA outperforms two classical resource discovery strategies as well as a popular neural network training algorithm.

1 Introduction

During recent years the use of peer-to-peer networks (P2P) has significantly increased and thus demand of high performance peer-to-peer networks is constantly growing. In order to obtain proper functioning of a P2P network a crucial point is to efficiently execute the P2P resource discovery, since an improper resource discovery strategy would lead to overwhelming query traffic and consequently to a waste of bandwidth for each single user.

This problem has been intensively analyzed and several solutions have been proposed in commercial packages and scientific literature. The solutions so far proposed can be classified into two categories: breadth-first search (BFS) and depth-first search (DFS). BFS strategies forward a query to multiple neighbors at the same time whereas DFS strategies forward only to one neighbor.

BFS strategies have been used in Gnutella, where the query is forwarded to all neighbors and the forwarding is controlled by a time-to-live parameter. This parameter is defined as the amount of hops required to forward the query. Two nodes are said to be n hops apart if the shortest path between them has length

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 61–70, 2007.
© Springer-Verlag Berlin Heidelberg 2007

n [1]. The main disadvantage of the Gnutella's mechanism is that it generates a massive traffic of query messages when the time-to-live parameter is high. In order to reduce query traffic, Lv et al. [2] proposed the *Expanding Ring*. This strategy establishes that the time-to-live parameter is gradually increased until enough resources have been found. Although use of the *Expanding Ring* is beneficial in terms of query packet reduction, it introduces some delay to resource discovery and thus implies a longer waiting time for the user. Kalogeraki et al. [3] and Menascé [4] proposed that only a subset of neighbors are selected randomly for forwarding. While in [3] a mechanism is proposed which stores the performance of the queries previously done for each neighbor and then uses this memory to direct subsequent queries, in [4] the earlier replies are cached in directory entries and queried prior to using broadcast probability. Yang and Garcia-Molina [1] proposed to heuristically select the first neighbor and further uses BFS for forwarding the query. In Gnutella2 a trial query is sent to the neighbors and estimates how widely the actual query should be forwarded.

In the DFS strategies, selection of the neighbor for query forwarding is performed by means of heuristics. Lv et al. [2] studied the use of multiple random walkers which periodically check the query originator in order to verify whether the query should be forwarded further. Tsoumakos and Roussopoulos [5] proposed using the feedback from previous queries in order to tune probabilities for further forwarding of random walkers. Crespo and Garcia-Molina [6] proposed routing indices, which provide shortcuts for random walkers in locating resources. Sarshar et al. [7] proposed replicating a copy of resources and thus ensure that resource discovery strategy locates at least one replica of the resource.

The main limitation of the previous studies, for both BFS and DFS strategies, is that all the approaches are restricted to only one search strategy. On the contrary, for the same P2P network, in some conditions it is preferable to employ both BFS and DFS strategies. In order to obtain a flexible search strategy, which intelligently takes into account the working conditions of the P2P network, Vapa et al. [8] proposed a neural network based approach (NeuroSearch) which adaptively combines BFS and DFS. In NeuroSearch, a trained neural network is able to map a specific input set to forward decisions in an if-then logic. Thanks to this logic, the resource discovery strategy can be applied also in devices with limited computing power. On the other hand, training neural networks to adapt to various conditions is challenging since it requires training in multiple topological scenarios thus leading to complicated computational requirements. It is therefore fundamental to investigate efficient training algorithms which lead to high performance in a short training time.

2 Problem Description

NeuroSearch [8] is a neural network-based approach which combines different local information units together as an input to multi-layer perceptron (MLP) neural network [9]. The neural network employed in NeuroSearch contains two hidden layers, both having 10 neurons and two different transfer functions in

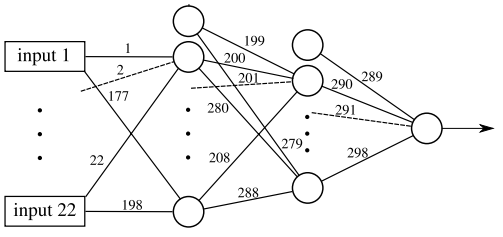


Fig. 1. MLP Neural Network

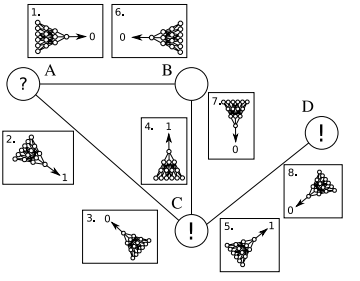


Fig. 2. Query Forwarding

hidden and output layers. The structure of this neural network (see Fig. 1) has been selected on the basis of previous studies carried out by means of the P2PRealm simulation framework [10]. Details regarding the functioning of this neural network are given in [8] and [10]. We characterize the query forwarding situation with a model consisting of 1) the previous forwarding node, 2) the currently forwarding node and 3) the receiver of the currently forwarding node. Upon receiving a query, the currently forwarding node selects the first of its neighbors and determines the inputs, related to that neighbor, of the neural network. The neural network output is then calculated. This output establishes whether or not the query will be forwarded to the neighbor. Next, all other neighbors including the previous forwarding node, are processed in a similar manner by means of the same neural network. Fig. 2, shows an example of the functioning of a P2P network with neural network based forwarding. The circles shown in the figure represent peers of the P2P network. The arcs between the peers represent the Transmission Control Protocol communication links between the peers. The rectangles represent a neural network evaluation for different neighbors. This paper addresses the problem in the training of a neural network (i.e. the determination of the set of weight coefficients W) of the kind in Fig. 1 with the aim summarized in Fig 2. As shown in Fig. 1, the weights can be divided into three categories on the basis of the layer to which they belong to. There are 22 input neurons and 10 neurons on both the hidden layers. Since one input is constant (*Bias*, see [8]) the total amount of weights is $22 * 9 + 10 * 9 + 10 = 298$. The weights can take values within the range $(-\infty, \infty)$. In order to estimate the quality of a candidate solution, the performance of the P2P network is analyzed with the aid of a simulator whose working principles are described in [10] and a certain number n of queries are performed. For each query, the simulator returns two outputs: the number of query packets P used in the query and the number of found resource instances R during the query. At each j^{th} query, these outputs are combined in the following way and F_j is determined:

$$F_j = \begin{cases} 0 & \text{if } P > 300 \\ 1 - \frac{1}{P+1} & \text{if } P \leq 300 \text{ AND } \quad R = 0 \\ 50 * R - P & \text{if } P \leq 300 \text{ AND } 0 < R < \frac{AR}{2} \\ 50 * \frac{AR}{2} - P & \text{if } P \leq 300 \text{ AND } \quad \frac{AR}{2} < R \end{cases} \tag{1}$$

In (1), the amount of Available Resources (AR) instances is constant at each query and the constant values 300 and 50 have been set according to the criterion explained in [8]. It must be noted that due to its formulation each F_j could likely contain several plateaus (see (1)). The total fitness over the n queries is given by $F = \sum_{j=1}^{n} F_j(W)$. It is important to remark that multiple queries ($n = 10$) are needed in order to ensure that the neural network is robust in different query conditions. The querying peer and the queried resource need to be changed to ensure that the neural network is not only specialized for searching resources from one part of the network or one particular resource alone. Therefore, two consecutive fitness evaluations do not produce the same fitness value for the same neural network. Since n queries are required and, for each query, the first forwarding node is chosen at random, fitness F is noisy. This noise is not Gaussian. Let us indicate with $PN(n)$ the distribution of this noise and thus formulate the optimization problem addressed in this paper:

$$\max (F(W) + Z) \text{ in } (-\infty, \infty)^{298}; Z \sim PN(n) \tag{2}$$

3 The Adaptive Global-Local Memetic Algorithm

In order to solve the problem in (2), the following Adaptive Global-Local Memetic Algorithm (AGLMA) has been implemented.

Initialization. An initial sampling made up of S^i_{pop} individual has been executed pseudo-randomly with a uniform distribution function over the interval $[-0.2, 0.2]$. This choice can be briefly justified in the following way. The weights of the initial set of neural networks must be small and comparable among each other in order to avoid one or a few weights dominating with respect to the others as suggested in [11], [12].

Parent Selection and Variation Operators. All individuals of the population S_{pop} undergo recombination and each parent generates an offspring. The variation occurs as follows. Associated with each candidate solution i is a self-adaptive vector h_i which represents a scale factor for the exploration. More specifically, at the first generation the self-adaptive vectors h_i are pseudo-randomly generated with uniform distribution within $[-0.2, 0.2]$ (see [11], [12]).

At subsequent generations each self-adaptive vector is updated according to [11], [12]:

$$h_i^{k+1}(j) = h_i^k(j)\, e^{(\tau N_j(0,1))} \quad \text{for} \quad j = 1, 2...n \tag{3}$$

where k is the index of generation, j is the index of variable ($n = 298$), $N_j(0,1)$ is a Gaussian random variable and $\tau = \frac{1}{\sqrt{2\sqrt{n}}} = 0.1659$. Each corresponding candidate solution W_i is then perturbed as follows [11], [12]:

$$W_i^{k+1}(j) = W_i^k + h_i^{k+1}(j)\, N_j(0,1) \quad \text{for} \quad j = 1, 2...n \tag{4}$$

Fitness Function. In order to take into account the noise, function F is calculated n_s times and an *Explicit Averaging* technique is applied [13]. More

specifically, each set of weights for a neural network (candidate solution) is evaluated by means of the following formula:

$$\hat{F} = F^i_{mean} - \frac{\sigma^i}{\sqrt{n_s}} \tag{5}$$

where F^i_{mean} and σ^i are respectively the mean value and standard deviation related to the n_s samples performed to the i^{th} candidate solution.

The penalty term $\frac{\sigma^i}{\sqrt{n_s}}$ takes into account distribution of the data and the number of performed samples [14]. Since the noise strictly depends on the solution under consideration, it follows that for some solutions the value of σ^i is relatively small (stable solutions) and so penalization is small. On the other hand, other solutions could be unstable and score 0 during some samples and give a high performance value during other samples. In these cases σ^i is quite large and the penalization must be significant.

Local Searchers. Two local searchers with different features in terms of search logic and pivot rule have been employed. These local searchers have the role of supporting the evolutionary framework, offering new search directions and exploiting the available genotypes [15].

1)Simulated Annealing. The Simulated Annealing (SA) metaheuristic [16] has been chosen since it offers an exploratory perspective in the decision space which can choose a search direction leading to a basin of attraction different from starting point W_0 and, thus, prevents an undesired premature convergence. The exploration is performed by using the same mutation scheme as was described in equations (3) and (4) for an initial self-adaptive vector h_0 pseudo-randomly sampled in $[-0.2, 0.2]$.

The main reason for employing the SA in the AGLMA is that the evolutionary framework should be assisted in finding better solutions which improve the available genotype while at the same time exploring areas of the decision space not yet explored. It accepts, with a certain probability, solutions with worse performance in order to obtain a global enhancement in a more promising basin of attraction. In addition, the exploratory logic aims to overcome discontinuities of the fitness landscape and to "jump" into a plateau having better performance. For these reasons the SA has been employed as a "global" local searcher.

2)Hooke-Jeeves Algorithm. The Hooke-Jeeves Algorithm (HJA) [17] is a deterministic local searcher which has a steepest descent pivot rule. The HJA is supposed to efficiently exploit promising solutions enhancing their genotype in a meta-Lamarckian logic and thus assist the evolutionary framework in quickly climbing the basin of attractions. In this sense the HJA can be considered as a kind of "local" local searcher integrated in the AGLMA.

Adaptation. In order to design a robust algorithm [15], at the end of each generation the following parameter is calculated:

$$\psi = 1 - \left| \frac{\hat{F}_{avg} - \hat{F}_{best}}{\hat{F}_{worst} - \hat{F}_{best}} \right| \tag{6}$$

where $\hat{F}_{worst}$, $\hat{F}_{best}$, and $\hat{F}_{avg}$ are the worst, best, and average of the fitness function values in the population, respectively. As highlighted in [18], ψ is a fitness-based measurement of the population diversity which is well-suited for flat fitness landscapes. The employment of this parameter, taking into account the presence of plateaus in the fitness landscape (i.e. areas with a very low variability in the fitness values.) ψ, efficiently measures the population diversity even when the range of variability of all fitness values is very small. The population has high diversity when $\psi \approx 1$ and low diversity when $\psi \approx 0$. A low diversity means that the population is converging (possibly in a suboptimal plateau). We remark that the absolute diversity measure used in [14], [19], [20] and [21] is inadequate in this case, since, according to this, the population diversity would be very low most of the time.

Coordination of the local searchers. The SA is activated by the condition $\psi \in [0.1, 0.5]$. This adaptive rule is based on the observation that for values of $\psi > 0.5$, the population diversity is high and therefore the evolutionary framework needs to have a high exploitation of the available genotypes (see [19], [18] and [21]). On the other hand, if $\psi < 0.5$ the population diversity is decreasing and application of the SA can introduce a new genotype in the population which can prevent a premature convergence. In this sense, the SA has been employed as a local searcher with "global" exploratory features. The condition regarding the lower bound of usability of the SA ($\psi > 0.1$) is due to the consideration that if $\psi < 0.1$ application of the SA is usually unsatisfactory since it most likely leads to a worsening in performance.

Moreover, the SA, in our implementation, is applied to the second best individual. This gives a chance at enhancing a solution with good performance without possibly ruining the genotype of the best solution. The initial temperature $Temp^0$ has been adaptively set $Temp^0 = \left|\hat{F}_{avg} - \hat{F}_{best}\right|$. This means that the probability of accepting a worse solution depends on the state of the convergence. In other words, the algorithm does not accept worse solutions when the convergence has practically occurred.

The HJA is activated when $\psi < 0.2$ and is applied to the solution with best performance. The basic idea behind this adaptive rule is that the HJA has the role of quickly improving the best solution while staying in the same basin of attraction. In fact, although evolutionary algorithms are efficient in detecting a solution which is near the optimum, they are not so efficient in "ending the game" of optimization. In this light, the action of the HJA can be seen as purely "local". The condition $\psi < 0.2$ means that the HJA is employed when there are some chances that optimal convergence is approaching. An early application of this local searcher can be inefficient since a high exploitation of solutions having poor fitness values would not lead to significant improvements of the population.

It should be noted that in the range $\psi \in [0.1, 0.2]$ both local searchers are applied to the best two individuals of the population. This range is very critical for the algorithm because the population is tending towards a convergence but still has not reached such a condition. In this case, there is a high risk of premature convergence due to the presence of plateaus and suboptimal basins of attraction

or false minima introduced by noise. Thus, the two local searchers are supposed to "compete and cooperate" within the same generation, merging the "global" search power of the SA and the "local" search power of the HJA. An additional rule has been implemented. When the SA has succeeded in enhancing the starting solution, the algorithm attempts to further enhance it by the application of the HJA under supervision of the evolutionary framework.

Dynamic population size in survivor selection. The population is resized at each generation and the S_{pop} individuals having the best performance are selected for the subsequent generation:

$$S_{pop} = S_{pop}^{f} + S_{pop}^{v} \cdot (1 - \psi)\,, \tag{7}$$

where S_{pop}^{f} and S_{pop}^{v} are the fixed minimum and maximum sizes of the variable population S_{pop}, respectively.

The dynamic population size has two combined roles. The first is to massively explore the decision space and thus prevent a possible premature convergence (see [19]), the second is to *Implicitly Average* in order to compensate for noise by means of the evaluations of similar individuals [13]. According to the first role, when $\psi \approx 0$ the population is converging and a larger population size is required to increase the exploration and possibly inhibit premature convergence by offering new search directions. On the other hand, if the population is spread out in the decision space it is highly desirable that the most promising solution leads the search and that the algorithm exploits this promising search direction. According to the second role, it is well-known that large population sizes are helpful in defeating the noise [22]. Furthermore, recent studies [14], [23] have noted that the noise jeopardizes functioning of the selection mechanisms especially for populations made up of individuals having similar performance, since the noise introduces a disturbance in pair-wise comparison. Therefore, the AGLMA aims to employ a large population size in critical conditions (low diversity) and a small population size when a massive averaging is unnecessary. The algorithm stops when either a budget condition on the number of fitness evaluations is satisfied or ψ takes a value smaller than 0.01.

4 Numerical Results

For the AGLMA 30 simulation experiments have been executed. Each experiment has been stopped after 1.5×10^6 fitness evaluations. At the end of each generation, the best fitness value has been saved. These values have been averaged over the 30 experiments available. The average over the 30 experiments defines the Average Best Fitness (ABF). Analogously, 30 experiments have been carried out with the Checkers Algorithm (CA) described in [11], [12] according to the implementation in [8], and the proposed here Adaptive Checkers Algorithm (ACA) which is the CA with the fitness as shown in (5) and the adaptive population size as shown in (7). For the same P2P network, the BFS according to the implementation in Gnutella and the random walker DFS proposed in [2]

have been applied. Table 1 shows the parameter settings for the three algorithms and the optimization results. The final fitness $\hat{F}^b$ obtained by the most successful experiment (over the 30 sample runs), the related number of query packets P used in the query and the number of found resource instances R during the query are given. In addition the average best fitness at the end of the experiments $< \hat{F} >$, the final fitness of the least successful experiment $\hat{F}^w$ and the related standard deviation are shown. Since the BFS follows a deterministic logic, thus only one fitness value is shown. On the contrary, the DFS under study employs a stochastic structure and thus the same statistic analysis as that of CA, ACA and AGLMA over 30 experiments has been carried out.

Table 1. Parameter setting and numerical results

PARAMETER	AGLMA	CA	ACA	BFS	DFS
EVOLUTIONARY FRAMEWORK					
S^i_{pop}	30	30	30	–	–
S_{pop}	$\in [20, 40]$	30	$\in [20, 40]$	–	–
sample size n_s	10	–	10	–	–
SIMULATED ANNEALING					
initial temperature $Temp^0$	adaptive	–	–	–	–
temperature decrease	hyperbolic	–	–	–	–
maximum budget per run	600	–	–	–	–
HOOKE-JEEVES ALGORITHM					
exploratory radius	$\in [0.5, 0.01]$	–	–	–	–
maximum budget per run	1000	–	–	–	–
NUMERICAL RESULTS					
P	350	372	355	819	514
R	81	81	81	81	81
$\hat{F}^b$	3700	3678	3695	3231	3536
$< \hat{F} >$	3654	3582	3647	–	3363
$\hat{F}^w$	3506	3502	3504	–	3056
std	36.98	37.71	36.47	–	107.9

Numerical results in Table 1 show that the AGLMA and ACA outperform the CA and that the AGLMA slightly outperformed the ACA in terms of the final solution found. Moreover, the AGLMA clearly outperforms the BFS employed in Gnutella and the DFS.

Figures 3 and 4 show the comparison of the performance. As shown, the AGLMA has a slower convergence than the CA and the ACA but reaches a final solution having better performance. It is also clear that the ACA has intermediate performance between the CA and AGLMA. The ACA trend, in early generations, has a rise quicker than the AGLMA but slower than the CA. On the other hand, in late generations, the ACA outperforms the CA but not the AGLMA. Regarding effectiveness of the noise filtering components, Fig. 4 shows that the ACA and the AGLMA are much more robust with respect to noise than the CA. In fact, the trend of the CA performance contains a high amplitude and

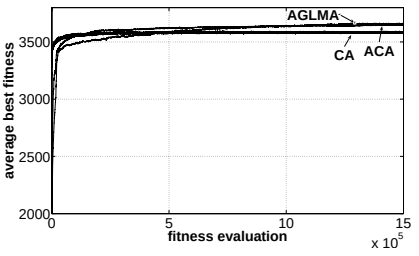


Fig. 3. Algorithmic Performance

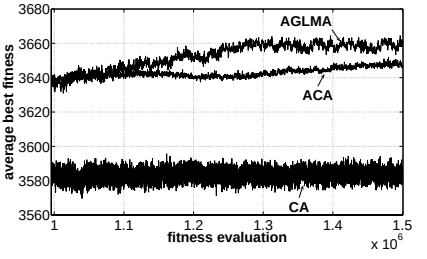


Fig. 4. Performance (zoom detail)

frequency ripple, while the ACA and AGLMA performance are roughly monotonic. Regarding effectiveness of the local searchers, the comparison between the ACA and the AGLMA shows that the AGLMA slightly outperforms the ACA tending to converge to a solution having a better performance.

5 Conclusion

This paper proposes an AGLMA for performing the training of a neural network, which is employed as computational intelligence logic in P2P resource discovery. The AGLMA employs averaging strategies for adaptively executing noise filtering and local searchers in order to handle the multivariate fitness landscape. These local searchers execute the global and local search of the decision space from different perspectives. The numerical results show that the application of the AGLMA leads to a satisfactory neural network training and thus to an efficient P2P network functioning. The proposed neural network along with the learning strategy carried by the AGLMA allows the efficient location of resources with little query traffic. Thus, with reference to classical resource discovery strategies (Gnutella BFS and DFS), the user of the P2P network obtains plentiful amounts of information about resources consuming a definitely smaller portion of bandwidth for query traffic. Regarding performance during the optimization process, comparison with a popular metaheuristic present in literature shows the superiority of the AGLMA in terms of final solution found and reliability in a noisy environment.

References

1. Yang, B., Garcia-Molina, H.: Improving search in peer-to-peer networks. In: Proc. of the 22nd Intern. Conf. on Distributed Computing Systems. (2002) 5–14
2. Lv, Q., Cao, P., Cohen, E., Li, K., Shenker, S.: Search and replication in unstructured peer-to-peer networks. In: Proc. of the 16th ACM Intern. Conf. on Supercomputing. (2002) 84–95
3. Kalogeraki, V., Gunopulos, D., Zeinalipour-Yazti, D.: A local search mechanism for peer-to-peer networks. In: Proc. 11th ACM Intern. Conf. on Information and Knowledge Management. (2002) 300–307

4. Menascé, D.A.: Scalable p2p search. IEEE Internet Computing **7**(2) (2003) 83–87
5. Tsoumakos, D., Roussopoulos, N.: Adaptive probabilistic search for peer-to-peer networks. In: Proc. 3rd IEEE Intern. Conf. on P2P Computing. (2003) 102–109
6. Crespo, A., Garcia-Molina, H.: Routing indices for peer-to-peer systems. In: Proc. of the 22nd IEEE Intern. Conf. on Distributed Computing Systems. (2002) 23–33
7. Sarshar, N., Boykin, P.O., Roychowdhury, V.P.: Percolation search in power law networks: Making unstructured peer-to-peer networks scalable. In: Proc. of the IEEE 4th Intern. Conf. on P2P Computing. (2004) 2–9
8. Vapa, M., Kotilainen, N., Auvinen, A., Kainulainen, H., Vuori, J.: Resource discovery in p2p networks using evolutionary neural networks. In: Intern. Conf. on Advances in Intelligent Systems - Theory and Applications, 067-04. (2004)
9. Engelbrecht, A.: Computational Intelligence-An Introduction. J. Wiley (2002)
10. Kotilainen, N., Vapa, M., Keltanen, T., Auvinen, A., Vuori, J.: P2prealm - peer-to-peer network simulator. In: IEEE Intern. Works. on Computer-Aided Modeling, Analysis and Design of Communication Links and Networks. (2006) 93–99
11. Chellapilla, K., Fogel, D.: Evolving neural networks to play checkers without relying on expert knowledge. IEEE Trans. Neural Networks, **10**(6) (1999) 1382–1391
12. Chellapilla, K., Fogel, D.: Evolving an expert checkers playing program without using human expertise. IEEE Trans. Evol. Computation, **5**(4) (2001) 422–428
13. Jin, Y., Branke, J.: Evolutionary optimization in uncertain environments - a survey. IEEE Transactions on Evolutionary Computation **9**(3) (2005) 303–317
14. Neri, F., Cascella, G.L., Salvatore, N., Kononova, A.V., Acciani, G.: Prudent-daring vs tolerant survivor selection schemes in control design of electric drives. In Rothlauf, F. et al., ed.: Applications of Evolutionary Computing, LNCS. Volume 3907., Springer (2006) 805–809
15. Krasnogor, N.: Toward robust memetic algorithms. In W. E. Hart et al., ed.: Recent Advances in Memetic Algorithms, Springer (2004) 185–207
16. Cerny, V.: A thermodynamical approach to the traveling salesman problem. Journal of Optimization, Theory and Applications **45**(1) (1985) 41–51
17. Hooke, R., Jeeves, T.A.: Direct search solution of numerical and statistical problems. Journal of the ACM, **8** (1961) pp. 212–229
18. Neri, F., Toivanen, J., Cascella, G.L., Ong, Y.S.: An adaptive multimeme algorithm for designing hiv multidrug therapies. IEEE/ACM Transactions on Computational Biology and Bioinformatics, Special Issue on Computational Intelligence Approaches in Computational Biology and Bioinformatics (2007) to appear.
19. Caponio, A., Cascella, G.L., Neri, F., Salvatore, N., Sumner, M.: A fast adaptive memetic algorithm for on-line and off-line control design of PMSM drives. IEEE Trans. on System Man and Cybernetics-part B 37(1) (2007) 28–41.
20. Neri, F., Toivanen, J., Mäkinen, R.A.E.: An adaptive evolutionary algorithm with intelligent mutation local searchers for designing multidrug therapies for HIV. Applied Intelligence, Springer (2007) to appear.
21. Neri, F., Mäkinen, R.A.E.: Hierarchical evolutionary algorithms and noise compensation via adaptation. In S. Yang et al., ed.: Evolutionary Computation in Dynamic and Uncertain Environments, Springer (2007) to appear.
22. Miller, B.L., Goldberg, D.E.: Genetic algorithms, selection schemes, and the varying effects of noise. Evolutionary Computation **4**(2) (1996) 113–131
23. Schmidt, C., Branke, J., Chick, S.E.: Integrating techniques from statistical ranking into evolutionary algorithms. In F. Rothlauf et al., ed.: Applications of Evolutionary Computing. Volume LNCS 3907., Springer (2006) 752–763

Evolutionary Computation for Quality of Service Internet Routing Optimization

Miguel Rocha[1], Pedro Sousa[1], Paulo Cortez[2], and Miguel Rio[3]

[1] Dep. Informatics / CCTC - Univ. Minho - Braga - Portugal
mrocha@di.uminho.pt, pns@di.uminho.pt
[2] Dep. Information Systems/ Algoritmi - Univ. Minho - Guimarães - Portugal
pcortez@dsi.uminho.pt
[3] University College London - London - UK
m.rio@ee.ucl.ac.uk

Abstract. In this work, the main goal is to develop and evaluate a number of optimization algorithms in the task of improving *Quality of Service* levels in TCP/IP based networks, by configuring the routing weights of link-state protocols such as OSPF. Since this is a complex problem, some meta-heuristics from the *Evolutionary Computation* arena were considered, working over a mathematical model that allows for flexible cost functions, taking into account several measures of the network behavior such as network congestion and end-to-end delays. A number of experiments were performed, resorting to a large set of network topologies, where *Evolutionary Algorithms (EAs)*, *Differential Evolution* and some common heuristic methods including *local search* were compared. *EAs* make the most promising alternative leading to solutions with an effective network performance even under unfavorable scenarios.

Keywords: Traffic Engineering, Quality of Service Routing, Evolutionary Algorithms, Differential Evolution, OSPF.

1 Introduction

The relevance of implementing *Quality of Service (QoS)* support mechanisms in IP-based networks has been fostered in the last few years by the integration of a number of new applications. Several distinct *QoS* aware architectures and traffic control mechanisms have been proposed in order to provide distinct service levels to networked applications [13]. In this context, *Internet Service Providers (ISPs)* have agreements with their clients and with other *ISPs* that have to be obeyed. To face such requirements, there is an important set of configuration tasks that have to be performed by network administrators in order to assure that correct resource provisioning is achieved in the domain.

There is not an unique solution to create a *QoS* aware infrastructure and any solution requires a number of components working together. However, independently of the particular solutions adopted that might be in place, there are components which have a crucial importance. One of such components has the ability to control the data path followed by packets traversing a given *Wide*

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 71–80, 2007.
© Springer-Verlag Berlin Heidelberg 2007

Area Network (WAN). In a *WAN*, consisting of a single administrative domain, there are alternative strategies for this purpose: Intra-domain routing protocols or *Multi-Protocol Label Switching (MPLS)* [2].

This work will focus on intra-domain routing protocols, and more specifically on the most commonly used today: *Open Shortest Path First (OSPF)* [12]. Here, the administrator assigns weights to each link in the network, which are used to compute the best path from each source to each destination node using the *Dijkstra algorithm* [5]. The results are then used to compute the routing tables at each node. Since in *OSPF* the weight setting process is the only way administrators can affect the network behavior, this choice is of crucial importance, having a major impact on network performance. Nevertheless, in practice, simple rules are typically used, like setting the weights inversely proportional to the link capacity, often leading to sub-optimal resource utilization.

An ideal way to improve the process of *OSPF* weight setting is to implement traffic engineering, assuming that the administrator has access to the traffic demands between each pair of nodes in the network. This was the approach taken by Fortz et al [7] where this task was viewed as an NP-hard optimization problem by defining a cost function that measures network congestion. Some local search heuristics have been proposed, as well as the use of meta-heuristics [6]. However, such approaches did not accommodate delay based constraints that are crucial to implement *QoS* aware networking services.

In this paper, a number of optimization algorithms (*Evolutionary Algorithms*, *Differential Evolution*, local search) are employed to calculate link-state routing weights, that optimize traffic congestion while simultaneously complying to specific delay requirements. A mathematical model of the problem that accommodates both congestion and delay constraints is used to define a bi-objective cost function and therefore to develop fitness functions for the algorithms, which are then used to calculate the optimal OSPF weights for each network link.

An important and direct outcome of the research work presented in this paper is the ability of developing network management tools which automatically provide network administrators with near-optimal routing configurations for *QoS* constrained networking scenarios. In this context, devising efficient and accurate routing optimization methods will be a major contribution for pursuing optimal routing configurations in the Internet.

2 Problem Description

The general routing problem [1] represents routers and links by a set of nodes (N) and arcs (A) in a directed graph $G = (N, A)$. In this model, c_a represents the capacity of each link $a \in A$. A demand matrix D is available, where each element d_{st} represents the traffic demand between nodes s and t. For each arc a, the variable $f_a^{(st)}$ represents how much of the traffic demand between s and t travels over arc a. The total load on each arc a (l_a) can be defined as:

$$l_a = \sum_{(s,t) \in N \times N} f_a^{st} \tag{1}$$

while the link utilization rate u_a is given by: $u_a = \frac{l_a}{c_a}$. It is then possible to define a congestion measure for each link ($\Phi_a = p(u_a)$) [7], using a penalty function p that has small values near 0, but as the values approach the unity it becomes more expensive and exponentially penalizes values above 1 (Figure 1).

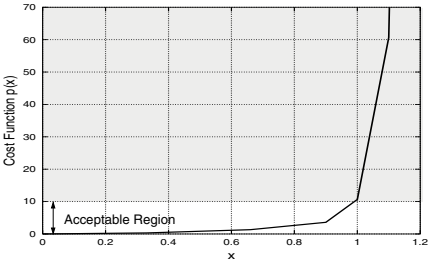


Fig. 1. Graphical representation of the penalty function p

In *OSPF*, all arcs have an integer weight. Every node uses these weights in the Dijkstra algorithm [5] to calculate the shortest paths to all other nodes in the network. All the traffic from a given source to a destination travels along the shortest path. If there are two or more paths with the same length, traffic is evenly divided among the arcs in these paths (load balancing) [10].

Let us assume a given solution, a weight assignment (w), and the corresponding utilization rates on each arc (u_a). In this case, the total routing cost is expressed by $\Phi(w) = \sum_{a \in A} \Phi_a(w)$ for the loads and corresponding penalties ($\Phi_a(w)$) calculated based on the given OSPF weights w. In this way, the *OSPF weight setting problem* is equivalent to finding the optimal weight values for each link (w_{opt}), in order to minimize the function $\Phi(w)$. The congestion measure can be normalized ($\Phi^*(w)$) over distinct topology scenarios and its value is in the range [1,5000]. It is important to note that when Φ^* equals 1, all loads are below 1/3 of the link capacity; in the case when all arcs are exactly full, the value of Φ^* is $10\frac{2}{3}$. This value will be considered as a threshold that bounds the acceptable working region of the network.

In order to include other *QoS* metrics, it was necessary to include delay constraints in this model. Delay requirements were modeled as a matrix DR, that for each pair of nodes $(s,t) \in N \times N$ gives the delay target for traffic between s and t (denoted by DR_{st}). In a way similar to the model presented before, a cost function was developed to evaluate the delay compliance for a solution, that takes into account the average delay of the traffic between the two nodes (Del_{st}), a value calculated by considering all paths between s and t with minimum cost and averaging the delays in each.

The *delay compliance ratio* for a given pair $(s,t) \in N \times N$ is, therefore, defined as $dc_{st} = \frac{Del_{st}}{DR_{st}}$. A penalty for delay compliance can be calculated using

function p. The γ_{st} function is defined according to $\gamma_{st} = p(dc_{st})$. This allows the definition of a delay cost function, given a set of *OSPF* weights (w):

$$\gamma(w) = \sum_{(s,t)\in N\times N} \gamma_{st}(w) \tag{2}$$

where the $\gamma_{st}(w)$ values represent the delay penalties for each end-to-end path, given the routes determined by the *OSPF* weight set w. This function can be normalized dividing the values by the sum of all minimum end-to-end delays to reach the value of $\gamma^*(w)$ (for each pair of nodes the minimum end-to-end delay is calculated as the delay of the path with minimum possible overall delay).

It is now possible to define the optimization problem addressed in this work. Indeed, given a network represented by a graph G, a demand matrix D and a delay requirements matrix DR, the aim is to find the set of *OSPF* weights w that simultaneously minimizes the functions $\Phi^*(w)$ and $\gamma^*(w)$. When a single objective is considered the cost of a solution w is calculated using functions $\Phi^*(w)$ for congestion and $\gamma^*(w)$ for delays. For multi-objective optimization a quite simple scheme was devised, where the cost of the solution is given by: $f(w) = \alpha\Phi^*(w) + (1-\alpha)\gamma^*(w)$. This scheme, although simple, can be effective since both cost functions are normalized in the same range.

3 Algorithms for OSPF Weight Setting

3.1 Evolutionary Algorithms

In this work, *Evolutionary Algorithms (EAs)* [9] are proposed to address the problems defined in the previous section, both by considering the single or the multi-objective formulation. In the proposed *EA*, each individual encodes a solution as a vector of integer values, where each value (gene) corresponds to the weight of an arc in the network (the values range from 1 to w_{max}). Therefore, the size of the individual equals the number of arcs in the graph (links in the network). The individuals in the initial population are randomly generated, with the arc weights taken from a uniform distribution in the allowed range.

In order to create new solutions, several reproduction operators were used, more specifically two mutation and one crossover operator:

- *Random Mutation*, replaces a given gene by a randomly generated value, within the allowed range;
- *Incremental/decremental Mutation*, replaces a given gene by the next or by the previous value (with equal probabilities) within the allowed range;
- *Uniform crossover*, a standard crossover operator [9].

In each generation, every operator is used to create new solutions with equal probabilities. The selection procedure is done by converting the fitness value into a linear ranking in the population, and then applying a roulette wheel scheme. In each generation, 50% of the individuals are kept from the previous generation, and 50% are bred by the application of the genetic operators. In the experiments a population size of 100 individuals was considered.

3.2 Differential Evolution

The *DE* method differs from the previous *EA* essentially in the reproduction operators. *DE* generates trial individuals by calculating vector differences between other randomly selected members of the population. In this work, a variant of the *DE* algorithm called *DE/rand/1* was considered that uses a binomial crossover [11]. In this case, the following scheme is followed for each individual i:

1. Randomly select 3 individuals r_1, r_2, r_3 distinct from i;
2. Generate a trial vector based on: $\boldsymbol{t} = \boldsymbol{r}_1 + F \cdot (\boldsymbol{r}_2 - \boldsymbol{r}_3)$
3. Incorporate coordinates of this vector with probability CR;
4. Evaluate the candidate and use it in the new generation if it is at least as good as the current individual.

Since *OSPF* weights are integer, it is necessary to round the values used in the *DE* before the evaluation. It is important to notice that in the *DE* all individuals in the population go throught the previous reproduction step. In the experiments, the population size was 20, F was set to 0.5 and CR to 0.6.

3.3 Local Search

A *local search (LS)* scheme was devised to improve the quality of a solution and works as follows: taking a set of weights w_i, a link is randomly selected to start the process. Firstly, it tries to increase the value of this weight by 1, if this implies that the solution is better. This process is repeated while the solution improves. If the first increase operation did not lead to a better solution, a decrease is tried and repeated while the solution improves.

The process is repeated for the next position, until all positions have been tested. The overall process is then repeated while the solution improves. Based on this *LS* operator, a *multi-start LS (MS-LS)* algorithm was devised: it starts with a random solution and applies the *LS* operator; this process is repeated and the best solution found is kept. The process is terminated when a maximum number of solutions has been evaluated.

3.4 Heuristic Methods

A number of heuristic methods were implemented [7] in order to assess the order of magnitude of the improvements obtained by the proposed methods when compared with the traditional weight setting heuristics, namely:

- **InvCap** - sets each link weight to a value inversely proportional to its capacity;
- **L2** - set each link weight to a value proportional to the its Euclidean distance;
- **Random** - a number of randomly generated solutions are analyzed and the best is selected.

4 Experiments and Results

In order to evaluate the proposed algorithms, a number of experiments was conducted. The experimental platform used in this work is presented in Figure 2. All the algorithms and the OSPF routing simulator were implemented using the *Java* language. The first step was the generation of a set of 12 networks by using the Brite topology generator [8], varying the number of nodes ($N = 30, 50, 80, 100$) and the average degree of each node ($m = 2, 3, 4$). This resulted in networks ranging from 57 to 390 links (graph edges). The link bandwidth was generated by an uniform distribution between 1 and 10 Gbits/s. The network was generated using the Barabasi-Albert model, using a heavy-tail distribution and an incremental grow type (parameters HS and LS were set to 1000 and 100, respectively). In all experiments only propagation delays were considered.

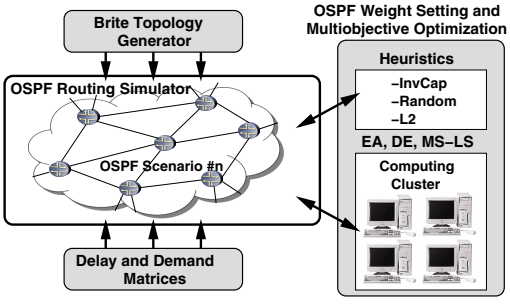


Fig. 2. Experimental platform for OSPF performance evaluation

Next, the demand and delay constraints matrices (D and DR) were generated. For each of the networks a set of three distinct D and DR matrices were created. A parameter (D_p) was considered, giving the expected mean of congestion in each link (u_a) (values for D_p in the experiments were 0.1, 0.2 and 0.3). For DR matrices, the strategy was to calculate the average of the minimum possible delays, over all pairs of nodes. A parameter (DR_p) was considered, representing a multiplier applied to the previous value (values for DR_p in the experiments were 3, 4 and 5). Overall, a set of $12 \times 3 \times 3 = 108$ instances of the optimization problem were considered.

The termination criteria of the optimization algorithms (*EAs*, *DE* and *LS*) was the maximum number of solutions evaluated that ranged from 50000 to 300000, increasing linearly with the number of links in the problem. The running times varied from a few minutes to a few hours, in the larger instances. In all cases, w_{max} was set to 20. For all the stochastic algorithms, 10 runs were executed in each case.

The results are grouped into two sets according to the cost function used. The first considers a single objective cost function, for the optimization of network congestion. The latter considers the case of a multi-objective cost function,

dealing with both congestion and delay optimization. In all figures presented in this section, the data was plotted in a logarithmic scale, given the exponential nature of the penalty function adopted.

4.1 Congestion

Since the number of performed experiments is quite high, it was decided to show only some aggregate results that can be used to draw conclusions. Table 1 shows the results for all the available networks, averaged by the demand levels (D_p), including in the last line the overall mean value for all problem instances. It is clear that the results get worse with the increase of D_p, as would be expected. Figure 3 plots the same results in a graphical way, showing in the the white area the acceptable working region, whereas an increasing level of gray is used to identify working regions with increasing levels of service degradation.

The comparison between the methods shows a superiority of the *EA*. In fact, the *EA* achieves solutions which manage a very reasonable behavior in all scenarios (worse case is 1.49). The heuristics manage very poorly, and even *InvCap*, an heuristic quite used in practice, gets poor results when D_p is 0.2 or 0.3, which means that the optimization with the *EAs* assures good network behavior in scenarios where demands are at least 200% larger than the ones where *InvCap* would assure similar levels of congestion. The results of *DE* and *MS-LS* are

Table 1. Results for the optimization of congestion (Φ^*) - averaged by demand levels

D_p	Random	EA	DE	MS-LS	L2	InvCap
0.1	75.75	1.02	1.02	1.12	215.94	1.50
0.2	498.74	1.18	1.41	1.50	771.87	57.70
0.3	892.87	1.73	3.64	6.08	1288.56	326.33
Overall	489.12	1.31	2.02	2.90	758.79	128.51

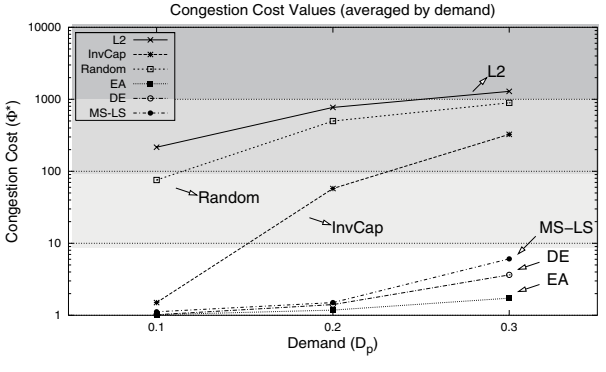


Fig. 3. Graphical representation of the results obtained by the different methods in congestion optimization (averaged by D_p)

acceptable, but nevertheless significantly worse than the ones obtained by the *EA*, and the gap increases with larger values of D_p.

4.2 Multi-objective Optimization

In this section, the results for the multi-objective optimization are discussed. The results are presented in terms of the values for the two objective functions (Φ^* and γ^*), since the value of f for these solutions can be easily obtained and are not relevant to the analysis. Given the space constraints only the value of 0.5 will be considered for parameter α, thus considering each aim to be of equal importance. Table 2 shows the results averaged by the demand level (D_p). From the table it is clear that the *EA* outperforms all other algorithms, followed by the *DE* and *MS-LS*. The heuristics behave quite badly, when both aims are taken into account. A similar picture is found looking at Table 3, where the results are averaged by the delay requirement parameter DR_p.

Table 2. Results for the multi-objective optimization - averaged by D_p

D	Random		EA		DE		MS-LS		L2		InvCap	
	Φ^*	γ^*	Φ^*	γ^*	Φ^*	γ^*	Φ^*	γ^*	Φ^*	γ^*	Φ^*	γ^*
0.1	88.00	106.79	1.17	1.92	1.18	2.04	1.73	4.07	215.94	1.76	1.50	260.30
0.2	481.50	136.68	1.47	2.32	1.65	2.92	3.38	8.30	771.87	1.76	57.70	260.30
0.3	949.85	148.96	2.41	3.23	4.58	5.64	15.31	15.95	1288.56	1.76	326.33	260.30
Overall	506.45	130.81	1.68	2.49	2.47	3.53	6.81	9.44	758.79	1.76	126.51	260.30

Table 3. Results for the multi-objective optimization - averaged by DR_p

DR	Random		EA		DE		MS-LS		L2		InvCap	
	Φ^*	γ^*	Φ^*	γ^*	Φ^*	γ^*	Φ^*	γ^*	Φ^*	γ^*	Φ^*	γ^*
3	535.28	283.16	1.95	4.22	2.78	6.42	9.65	21.29	758.79	2.94	128.51	577.94
4	505.69	82.04	1.59	1.78	2.44	2.36	6.12	4.65	758.79	1.25	128.51	158.85
5	478.37	27.23	1.51	1.48	2.38	1.82	4.65	2.39	758.79	1.10	128.51	44.13

A different view is offered by Figures 4 and 5 where the results are plotted with the two objectives in each axis. The former shows the results averaged by the demand levels and the latter by the delay requirements parameter. In these graphs, the good overall network behavior of the solutions provided by the *EA* is clearly visible, both in absolute terms, regarding the network behavior in terms of congestion and delays, and when compared to all other alternative methods. In fact, it is easy to see that no single heuristic is capable of acceptable results in both aims simultaneously. *L2* behaves well in the delay minimization but fails completely in congestion; *InvCap* is better on congestion (although in a very limited range) but fails completely in the delays. *DE* gets results that are in an acceptable range, but are always significantly worse than those of the *EAs*, and *MS-LS* does not manage good results when the problem instances get harder.

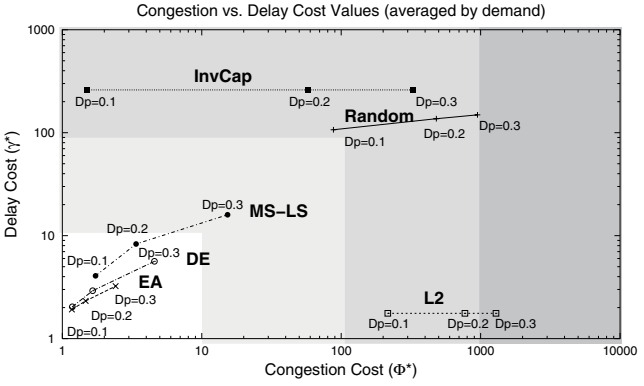


Fig. 4. Graphical representation of the results obtained by the different methods in the multi-objective optimization (averaged by D_p)

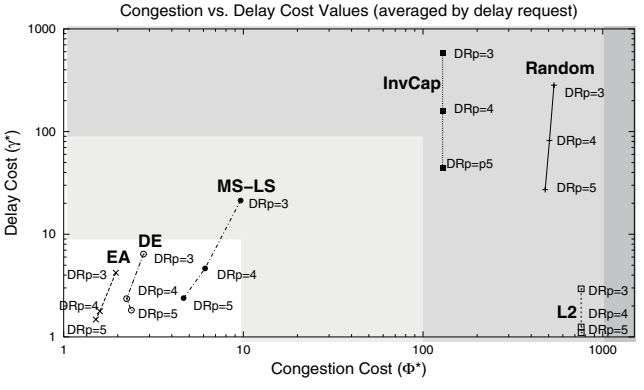


Fig. 5. Graphical representation of the results obtained by the different methods in the multi-objective optimization (averaged by DR_p)

5 Conclusions and Further Work

The optimization of OSPF weights brings important tools for traffic engineering, without demanding modifications on the basic network management model. This work presented *Evolutionary Computation* approaches for multi-objective routing optimization in the Internet. Resorting to a set of network configurations, each constrained by bandwidth and delay requirements, it was shown that the proposed *EAs* were able to provide OSPF weights that can lead to good network behavior. The performance of *EAs* was compared with other algorithms (*DE*, local search, heuristics) clearly showing its superiority. The proposed optimization framework, although requiring some computational effort, can be achieved in useful time and implemented in a real-world scenario.

Although a simple weighting method was used to face the multi-objective nature of the problem, the results were of high quality. This is probably due to the effort of normalizing both cost functions. Nevertheless, the consideration of specific *EAs* to handle this class of problems [4] will be taken into account in future work. *Memetic Algorithms*, that consider local optimization procedures embedded in the *EA*, have also been attempted in the congestion optimization problem [3]. Their application in this bi-objective scenario is also a research direction that has a strong potential.

References

1. R. K. Ahuja, T. L. Magnati, and J. B. Orlin. *Network Flows.* Prentice Hall, 1993.
2. D. Awduche and B. Jabbari. Internet traffic engineering using multi-protocol label switching (MPLS). *Computer Networks*, 40:111–129, 2002.
3. L. Buriol, M. Resende, C. Ribeiro, and M. Thorup. A hybrid genetic algorithm for the weight setting problem in OSPF/IS-IS routing. *Networks*, 2003.
4. C.A. Coello Coello. *Recent Trends in Evolutionary Multiobjective Optimization*, pages 7–32. Springer-Verlag, London, 2005.
5. E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(269-271), 1959.
6. M. Ericsson, M. Resende, and P. Pardalos. A Genetic Algorithm for the Weight Setting Problem in OSPF Routing. *J. Combinatorial Optimiz.*, 6:299–333, 2002.
7. B. Fortz and M. Thorup. Internet Traffic Engineering by Optimizing OSPF Weights. In *Proceedings of IEEE INFOCOM*, pages 519–528, 2000.
8. A. Medina, A. Lakhina, I. Matta, and J. Byers. BRITE: Universal Topology Generation from a User's Perspective. Technical Report 2001-003, 2001.
9. Z. Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs.* Springer-Verlag, USA, third edition, 1996.
10. J. Moy. *OSPF, Anatomy of an Internet Routing Protocol.* Addison Wesley, 1998.
11. R. Storn and K. Price. Differential Evolution - a Simple and Efficient Heuristic for Global Optimization over Continuous Spaces. *Journal of Global Optimization*, 11:341–359, 1997.
12. T.M. ThomasII. *OSPF Network Design Solutions.* Cisco Press, 1998.
13. Zheng Wang. *Internet QoS: Architectures and Mechanisms for Quality of Service.* Morgan Kaufmann Publishers, 2001.

BeeSensor: A Bee-Inspired Power Aware Routing Protocol for Wireless Sensor Networks

Muhammad Saleem and Muddassar Farooq

Centre for Advanced Studies in Engineering,
Islamabad 44000, Pakistan
{msaleem,mfarooq}@case.edu.pk

Abstract. Wireless Sensor Networks (WSNs) are becoming an active area of research. They consist of small nodes with limited sensing, computation and wireless communication capabilities. The success of WSNs in real world applications is primarily dependent on a key requirement: ability to provide a communication infra-structure for dissemination of sensed data to a sink node in an energy efficient manner. Therefore, in this paper we propose a bee-inspired power aware routing protocol, *BeeSensor*, that utilizes a simple bee agent model and requires little processing and network resources. The results of our extensive experiments demonstrate that *BeeSensor* delivers better performance in a dynamic WSNs scenario as compared to a WSN optimized version of Adhoc On-demand Distance Vector (*AODV*) protocol while its computational and bandwidth requirements are significantly smaller.

1 Introduction

Wireless sensor networks are becoming an active area of research due to their expected key role in diverse real world civil and military applications [8]. The spectrum of applications includes target field imaging, intrusion detection, weather monitoring, security and tactical surveillance and disaster management [1][8]. WSNs are created in an adhoc fashion through the wireless communication interfaces of sensor nodes when few hundreds or thousands of them are scattered in a geographical area. But each node itself has limited hardware resources [1][8]. Such resource constrained sensor nodes put a challenge on the routing infrastructure: power aware routing protocols with low processing complexity and small bandwidth utilization by control messages have to be designed to deliver the sensed data to a sink node. Moreover such routing protocols have to be self-organizing without a central controller due to the unpredictable environment in which they are expected to be deployed. Last but not least they have to be scalable, robust and performance efficient [1] with an ability to keep the network operational for an extended period of time.

In comparison, Nature inspired routing protocols are also gaining popularity because they do not require a priori global system model of the network. Rather they utilize a local system model as observed by the agents. The agents gather

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 81–90, 2007.
© Springer-Verlag Berlin Heidelberg 2007

the network state in a decentralized fashion and leave the corresponding information on visited nodes. This information enables them to make routing decisions in a distributed way without having the knowledge of complete network topology. The algorithms can adapt autonomously to changes in the network, or in traffic patterns. Consequently, such algorithms are self-organizing, decentralized and simple. *AntNet* [3], *BeeHive* [10] and their variants are becoming state-of-the-art Nature inspired routing algorithms. However, their application for routing in WSNs has received little attention. Some preliminary efforts reported in [13] are limited to improving basic *AntNet* for performance enhancement in WSNs. An ant-based routing protocol based on Ant Colony Optimization (ACO) metaheuristics is presented in [2]. The reported results indicate that the average remaining energy at the end of experiments is higher than few of its other variants. Apart from these sparse attempts to utilize Nature inspired routing protocols in WSNs, majority of routing protocols for WSNs are designed on the basis of classical routing techniques. Sensor Protocols for Information via Negotiation (SPIN) [5] is a data centric protocol in which energy efficient routing is done through the negotiation of high level meta-data descriptors. Directed diffusion [6] is a popular routing paradigm which introduces the idea of aggregating the data coming from different sources to eliminate data redundancy. A number of energy efficient variants of directed diffusion are surveyed in [1]. Power-Efficient Gathering in Sensor Information Systems (PEGASIS) [7] is a variant of Low Energy Adaptive Clustering Hierarchy (LEACH) [4] protocol in which sensor nodes are partitioned into distinct clusters in an energy efficient manner. Geographic and Energy Aware Routing (GEAR) [11] uses geographical information to route events toward the sink node.

The major contribution of the work presented in this paper is a simple, scalable, self-organizing, power aware and performance optimized bee-inspired routing protocol for WSNs. The algorithm has been carefully engineered by taking inspiration from relevant features of *BeeAdHoc* [9] and *BeeHive* [10] protocols which are inspired from the foraging principles of honey bees. *BeeHive* delivers better performance with a simple agent model in fixed networks as compared to existing algorithms while *BeeAdHoc* delivers similar or better performance as compared to other adhoc routing algorithms (AODV, DSR) but at least energy cost. As a result, *BeeSensor* achieves better performance with little energy consumption as compared to a WSN optimized *AODV* protocol.

Organization of Paper. The rest of the paper is organized as follows. In Section 2.1, a brief overview of *bee agent model* that is the core component of *BeeSensor* protocol is provided. The protocol itself is described in Section 2.2. Section 3 contains the description of our experimental framework and definitions of relevant performance metrics that are used to compare *BeeSensor* with a WSN optimized *AODV* protocol. The results obtained from comprehensive experiments are discussed in Section 4. Finally, we conclude the paper by showing that *BeeSensor* not only performs better than *AODV*, but its processing and bandwidth costs are also significantly smaller.

2 BeeSensor: Architecture and Working

2.1 BeeSensor Agent's Model

BeeSensor works with three types of agents: packers, scouts and foragers.

Packers. Packers receive data packets from an application layer and locate an appropriate forager (see below) for them at the source node. At sink node, they recover data from the payload of foragers and provide it to the application layer.

Scouts. Scouts in *BeeSensor* are classified as *forward scouts* and *backward scouts* depending upon the direction in which they travel. A source node that detects an event launches a *forward scout* when it does not have route to a sink node. A *forward scout* is propagated using the broadcasting principle to all neighbors of a node. Each *forward scout* has a unique id and it also carries the detected event in its payload. In *BeeSensor*, scouts do not construct a source header in which complete sequence of nodes up to destination is saved. As a result, scouts have got a fixed size that is independent of the path length (number of hops) between the source and a sink node. Moreover, the forward and return paths of a scout may not necessarily be the same. Once *forward scout* reaches at the sink node, it starts its return journey as a *backward scout*. When this *backward scout* arrives at the source node, a dance number is calculated by using the minimum remaining energy of the path that indicates the number of foragers to be cloned from this scout.

Foragers. In *BeeSensor*, foragers are the main workers which transport data packets from a source node to the sink node. They receive data packets from packers at a source node and deliver them to the packers at the sink node. The basic motivation in *BeeSensor* is to discover multiple paths to a sink node, therefore, each forager in addition to its forager id gets a unique path ID (PID) as well. A forager follows point-to-point mode of transmission until it reaches the sink node. A forager is piggy backed to the source node in the Acknowledgement (ACKs) of data packets generated by a lower data link layer. As a result, the overhead of sending foragers using explicit network layer transmissions is avoided. Once foragers arrive at the source node, a dance number is calculated in a similar way as in case of a *backward scout*. Consequently, more packets are routed through better quality paths. Remember foragers in *BeeSensor* do not carry complete source route in their header. Rather small forwarding tables at intermediate nodes are used to simulate the source routing behavior. The reason for this significant modification is that source routing based protocols do not scale for large networks because size of foragers is directly dependent on the length of the path between a source and a sink node.

2.2 Protocol Description

BeeSensor works in four distinct phases: Scouting, foraging, multiple path discovery and swarming. We discuss each of them separately as under.

Scouting. When a source node detects an event and is unable to locate any forager to carry the event to its sink node, scouting process is initiated. The node generates a *forward scout*, puts the event in the payload of this scout and broadcasts it to all its neighbors. When an intermediate node within two hops radius of the source node receives the *forward scout*, it updates its scout cache, increments the hop count and rebroadcasts it. Intermediate nodes more than two hops away stochastically decide whether to further rebroadcast a scout or not shown by dashed lines in Fig. 1. We verified that the *stochastic broadcasting* always delivers the scouts to the sink node launched by the most distant source node in large topologies (400 nodes). If another replica of an already broadcasted scout is received via another path at an intermediate node, its information is stored in scout cache and then it is dropped. It is worth mentioning here that the scouts launched by a source node do not search for a particular destination rather they carry information about the generated event. Any sink node interested in the events can respond by sending a *backward scout* to the source node. If a source node does not receive a *backward scout* after scouting a certain number of times, the scouting process is stopped indicating that no node is interested in the events. The nodes that do not broadcast a forward scout cannot lie on the route and hence are not required to remain active.

When a sink node receives the *forward scout*, it first updates its scout cache, then it transforms the *forward scout* into a *backward scout* and assigns it a unique path ID (PID). The sink node can get different replicas of the same scout through different neighbors and it will return a *backward scout* for each replica. This is an indication that a sink node is reachable from the source node via different neighbors of the sink node. We also associate a cost C_{ns} with each of the neighbors from which *forward scout* is received and it is defined as:

$$C_{ns} = E_r^n - \frac{H_{ns}}{K} \tag{1}$$

Where C_{ns} is the cost of selecting neighbor n for reaching the source node s, E_r^n is the remaining energy of node n, H_{ns} is the numbers of hops to reach the source node through n and K is a weighting factor. Higher values of K lower the significance of the path length (H_{ns}). The sink node forwards *backward scout* to the neighbor available in the scout cache for which this cost is maximum.

When a subsequent node receives this *backward scout*, it again selects the best neighbor in a similar way as done by the sink node. Then it updates its forwarding table which contains three fields (see Fig. 1): ID of the node from which it got the *backward scout* is NextHop, ID of the node to which the *backward scout* is to be forwarded is PrevHop and PID is the path ID. Then it updates the minimum remaining energy field of the *backward scout* and forwards it to the selected neighbor. This process is repeated at all intermediate nodes until the *backward scout* is back at the source node. As already mentioned, the source evaluates quality of the path by calculating the value of dance number which depends on the minimum remaining energy of the path reported by the *backward scout*. Finally it updates the routing table. A routing table entry consists of six fields (see Fig. 1): ID of the source node, a unique message ID generated at the application layer of the source

(these two fields uniquely identify the source of events), destination ID, path ID, next hop and number of recruited foragers.

Foraging. Source nodes also maintain a small event cache in which the events, generated during the route discovery process, are stored. Once the route is discovered, routing of events to the sink nodes is started. The source node selects an appropriate forager from the routing table, adds the event in the payload of the forager and forwards it to the next hop. Each forager has a path ID that determines which route, decided at the source node, it will follow to reach the sink node. When an intermediate node receives this forager, it forwards it to the next hop using the PID. For example, when node 2, shown in Fig. 1, receives a forager with PID=1, it sends it to the node 3. All subsequent nodes keep switching this forager until it reaches the sink node.

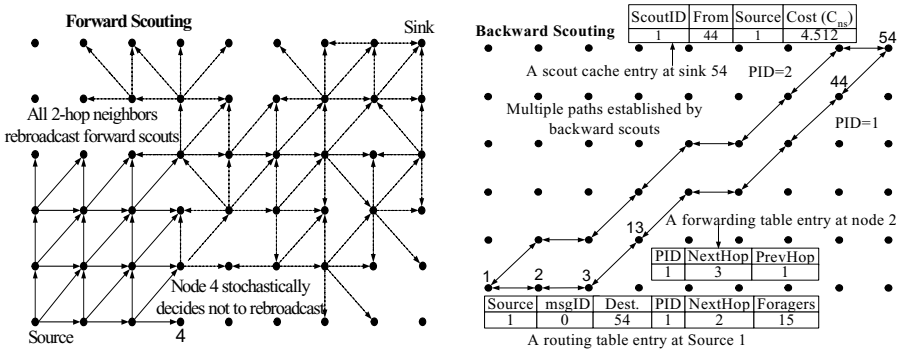


Fig. 1. Forward and backward scouting in *BeeSensor*

Multipath discovery. *BeeSensor* does not establish multiple paths during the initial route discovery phase. Rather a more conservative approach is adopted in which the sink node initially generates a *backward scout* and then waits for the events to start flowing towards it. After receiving a certain number of events, the sink node launches another *backward scout* with a unique PID using the information in its scout cache which is transported back to the source node in a similar way as the initial one. This *backward scout* is most likely to follow a different path as the path with PID=2 shown in Fig. 1. In this way *BeeSensor* avoids the overhead of maintaining multiple paths for applications that only generate data occasionally. However, for applications generating continuous traffic, multiple paths are established and traffic load is distributed accordingly. But maintenance of multiple paths costs additional processing and bandwidth.

Swarming. *BeeSensor* does not use explicit swarm bees to return the foragers back to the source nodes. Rather, it piggybacks the foragers to the source node in the ACKs of data packets generated by a lower data link layer. Swarming is helpful in verifying the validity of routes. If a source does not get back the foragers from a particular path, it is assumed that the path to the sink node is

lost and the corresponding entry from the routing table is deleted. If all paths are lost, scouting process is restarted on the arrival of next event. Similarly, if routing entries at the source node or forwarding table entries at intermediate nodes are not used for a certain period of time, they are also invalidated. As a result, no explicit special messages are needed to check the validity of links and to inform other nodes if they have become invalid. *BeeSensor* also supports explicit swarming and PrevHop entries in the forwarding tables may be used for this purpose. In addition, sink nodes must also maintain the forwarding table to route swarms bees back to the source node using the PrevHop field. In *BeeSensor*, only those nodes must remain active that have valid routing information.

3 Empirical and Performance Evaluation Framework

We used Prowler [12], a probabilistic wireless sensor network simulator, for conducting comprehensive empirical evaluation of *BeeSensor*. Prowler provides simple yet realistic radio and Media Access Control (MAC) models for the Berkeley mote platform. It also supports an event-based structure similar to TinyOS/NesC that makes it easy to deploy an algorithm on real hardware sensor nodes. We utilized the features of RMASE [12] framework, which is implemented as an application in Prowler, for realizing *BeeSensor*.

BeeSensor was compared with an energy optimized version of *AODV* that is distributed with RMASE framework. We selected *AODV* for comparison because it, like *BeeSensor*, discovers routes on demand only. It relies on the feedback of a lower layer for detecting broken links avoiding the use of explicit HELLO packets. In addition, intermediate nodes do not generate reply (RREP) to a route request (RREQ). This version of *AODV* also employs cross layer techniques to remove paths that have high packet loss. Consequently, its performance is improved.

We evaluated both protocols, *BeeSensor* and *AODV*, in two operational scenarios: first was a static scenario in which source and sink nodes were stationary and in second scenario source nodes were changing dynamically. The static scenario was tested on a sensor network of 20, 30 and 40 nodes placed randomly in the sensing field. Both algorithms achieved success rate of 95% in this simple scenario while rest of the performance parameters showed a similar trend as in case of the second scenario. Therefore, we skip these results for the sake of brevity.

We decided to use a real world example of target tracking for the second scenario. In target tracking, sensor nodes in the vicinity of a moving target generate some arbitrary sequence of events depending upon the location of the target. These events have to be delivered to a sink node for further analysis. Such a dynamic environment puts challenging demands on the routing protocols.

In this scenario, we placed the sensor nodes on a square grid and did a comprehensive evaluation of both protocols on 9, 16, 25, 49, 64 and 72 nodes for 300 seconds. We assumed symmetric links between nodes. Initial energy level of the nodes was set to 5 Joules (J). All reported values are an average of the values obtained from five independent runs. Choice of five runs is based on our

observation that ten runs in case of the static scenario gave the same results as obtained from five runs. Although a number of performance metrics are measured, but due to page limitations, we report the following:

Latency. It is defined as the difference in time when an event is generated at a source and when it got delivered at the sink node. We report average latency.

Packet delivery ratio. It is the ratio of total number of events received at a sink node to the total number of events generated by the source nodes in the sensor network (an event is dispatched in a packet).

Energy consumption. We report total energy consumed by the sensor nodes in a network during each experiment.

Energy efficiency. It is defined as the total energy consumed per 1000 bits delivered in the network (J/Kbits).

Lifetime. It is defined as the difference of average remaining energy levels of sensor nodes and their standard deviation. The basic motivation behind this definition is that an algorithm should try to maximize average remaining energy levels of nodes with a small standard deviation. We report it in percentage (%).

Cyclic complexity. It is defined as the total number of CPU cycles consumed, for processing control packets and forwarding data packets, by a protocol during an experiment per 1000 bits of data delivered at the destination (Cycles/Kbits).

Control overhead. It is defined as the number of control packets generated in the network per 1000 bits of delivered data (packets/Kbits).

4 Results

Latency and packet delivery ratio. Fig. 2 shows the packet delivery ratios and latencies of both protocols. *BeeSensor* consistently delivers more than 85% packets but the value for *AODV* ranges from 20% to approximately 65%. In 9 nodes experiment, the difference in packet delivery ratio of both protocols is about 65%. On smaller topologies, tracking application allows sensor nodes to transmit short bursts of events only and then stop. If a routing protocol is unable to find a path within this short interval, it simply loses the events generated in that interval. *AODV* has unstable behavior due to pure flooding of RREQ packets, therefore, it could not quickly discover new routes resulting in a bad performance. *BeeSensor* achieves better packet delivery ratio as compared to *AODV* due to four important features: first, it makes use of restrictive flooding which results in quick convergence of the protocol; second, it launches a scout carrying the first generated event and most of the time it is able to find routes in the first attempt; third, it maintains a small event cache to queue events while route discovery is in progress; fourth, it utilizes a simple packet switching model in which intermediate nodes do not perform complex routing table lookup as in *AODV*. Rather, they switch packets using a simple forwarding table at a faster rate. However, this significant improvement in packet delivery ratio of

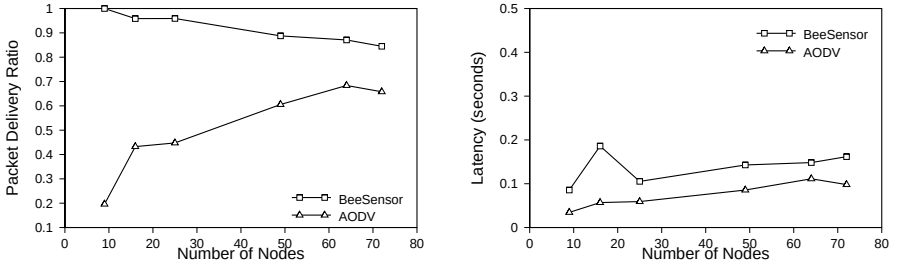


Fig. 2. Latency and packet delivery ratio

BeeSensor comes at a slightly higher latency as compared to *AODV*. This is because *BeeSensor* distributes packets on multiple paths rather than sending them on shortest path like *AODV* does. Moreover, it adaptively routes packets according to the remaining energy of the nodes of a path and the length of the path in terms of number of hops. Note that the difference in latency on larger sensor networks is relatively smaller as compared to smaller ones.

Cyclic complexity and control overhead. Fig. 3 shows cyclic complexities and control overheads of both protocols per 1000 bits of delivered data. It is evident from the results that *BeeSensor* has a significantly smaller cyclic complexity and control overhead which is about 50% less as compared to *AODV*. The reasons, already discussed in the previous paragraphs, are: simple agent model, an efficient packet switching algorithm that utilizes small forwarding tables stored on intermediate nodes. Complex routing table lookup is only performed at the source of the event (see Section 2.2). In comparison, *AODV* performs this complex routing tables search at each intermediate node.

We observed that the control overhead of *AODV* is significantly large as compared to *BeeSensor* due to its approach of pure flooding of RREQ packets. This makes network unstable and as a consequence RREP packet is lost most of the time which results in resending RREQ packet. In comparison, in *BeeSensor* only first two hops neighbors do pure flooding while rest of the nodes do probabilistic

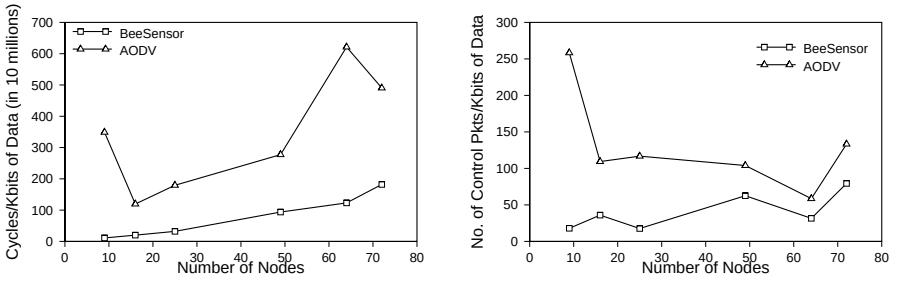


Fig. 3. Cyclic complexity and control overhead per Kbits

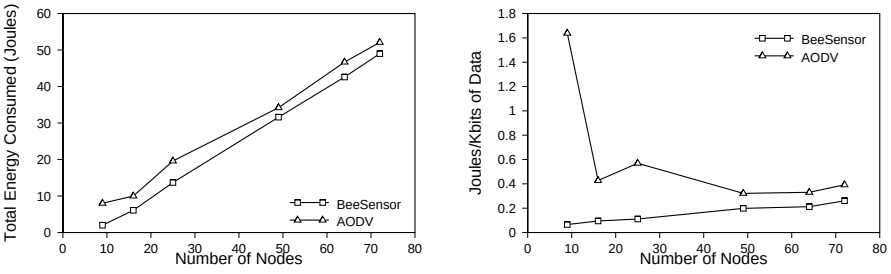


Fig. 4. Total energy consumption and energy efficiency

rebroadcast. This approach not only generates less control traffic but also successfully and quickly discovers the routes.

Total energy and energy efficiency. Energy consumption is an important performance metric that is of concern in wireless sensor networks. Fig. 4 shows the total energy consumption in the network along with energy efficiency. It is clear from Fig. 4 that *BeeSensor* not only consumes less energy but also has better energy efficiency than *AODV*. It is interesting to note that *BeeSensor* consumes 5-10% less energy even though it delivered 20-64% additional data which ultimately results in a better energy efficiency (J/Kbits).

Lifetime. Lifetime of the network at the end of each experiment is listed in Table 1. The results clearly show that *BeeSensor* has significantly better lifetimes than *AODV*. The better lifetime is because of *BeeSensor's* ability to avoid low energy nodes during route discovery. In comparison *AODV* always prefers shortest path. It is worth mentioning that the reasonable lifetime achieved by *AODV* is not due to its even distribution of data packets across the network but due to pure flooding of RREQ packets which by nature evenly depletes the energy resources. In contrast, *BeeSensor* consumes majority of its energy during data transmission and hence its better lifetime is due to the fact that it distributes the data traffic according to the current energy levels of intermediate nodes.

Table 1. Lifetime of the network

Algorithms	9 nodes	16 nodes	25 nodes	36 nodes	49 nodes	72 nodes
BeeSensor	90	83.3	82	75.14	75	77
AODV	53.55	65.53	70.5	69.14	74	71.7
Difference (%)	68	27	16	6	8.6	7.3

5 Conclusion and Future Work

In this paper we have proposed a new bee-inspired power aware routing protocol for wireless sensor networks, *BeeSensor*, which delivers superior performance as

compared to a WSN optimized version of *AODV*. The superior performance is achieved with a simple agent model that requires significantly little processing and network resources. We realized *BeeSensor* by using RMASE framework in Prowler simulator. *BeeSensor* delivered approximately 85% packets as compared to 60% of *AODV*. Similarly, computational complexity and control overhead of *BeeSensor* was significantly smaller as compared to *AODV*. Lifetime achieved by *BeeSensor* is also larger than *AODV*. Our future efforts are focused in analyzing the scalability of *BeeSensor* for large sensor area networks and comparing it with recently proposed other Nature inspired routing protocols.

References

1. J.N. Al-Karaki and A.E. Kamal. Routing techniques in wireless sensor networks: A survey. *IEEE Wireless Communications*, 11(6):6–28, Dec. 2004.
2. T. Camilo, C. Carreto, J. S. Silva, and F. Boavida. An energy-efficient ant-based routing for wireless sensor networks. In *Int. Workshop on Ant Colony Optimization and Swarm Intelligence, ANTS 2006, Sep.*, 2006.
3. G. D. Caro and M. Dorigo. AntNet: Distributed stigmergetic control for communication networks. *Journal of Artificial Intelligence Research*, 9:317–365, Dec. 1998.
4. W.R. Heinzelman, A. Chandrakasan, and H. Balakrishnan. Energy-efficient communication protocol for wireless microsensor networks. In *Proceedings of 33rd Hawaii Intl. Conference on System Sciences, Jan.*, 2000.
5. W.R. Heinzelman, J. Kulik, and H. Balakrishnan. Adaptive protocols for information dissemination in wireless sensor networks. In *Proceedings of MobiCom 99, Seattle, WA, Aug.*, 1999.
6. C. Intanagonwiwat, R.Govindan, D.Estrin, J. Heidemann, and F. Silva. Directed diffusion for wireless sensor networking. *IEEE/ACM Transactions on Networking*, 11(1), Feb. 2003.
7. S. Lindsey and C.S. Raghavendra. Pegasis: Power-efficient gathering in sensor information systems. In *Proceedings of IEEE Aerospace Conf., March*, 2002.
8. C.S. Raghavendra, K.M.S. Krishna, and T. Znati, editors. *Wireless Sensor Networks.* Springer-Verlag, 2004.
9. H.F. Wedde, M. Farooq, T. Pannenbaecke, B. Vogel, C. Mueller, J. Meth, and R. Jeruschkat. Beeadhoc: an energy efficient routing algorithm for mobile ad hoc networks inspired by bee behavior. In *Proceedings of conference on Genetic and evolutionary computation GECCO '05, June*, 2005.
10. H.F. Wedde, M. Farooq, and Y. Zhang. BeeHive: An efficient fault-tolerant routing algorithm inspired by honey bee behavior. In *Ant Colony Optimization and Swarm Intelligence, LNCS 3172*, pages 83–94. Springer Verlag, Sep. 2004.
11. Y. Yu, D. Estrin, and R. Govindan. Geographical and energy-aware routing: A recursive data dissemination protocol for wireless sensor networks. UCLA Computer Science Department Technical Report, UCLA-CSD TR-01-0023, May, 2001.
12. Y. Zhang. Routing modeling application simulation environment. http://www2.parc.com/isl/groups/era/nest/Rmase/.
13. Y. Zhang, L.D. Kuhn, and M.P.J. Fromherz. Improvements on ant routing for sensor networks. In *Int. Workshop on Ant Colony Optimization and Swarm Intelligence, ANTS. 2004, Sep.*, 2004.

Radio Network Design Using Population-Based Incremental Learning and Grid Computing with BOINC

Miguel A. Vega-Rodríguez, David Vega-Pérez, Juan A. Gómez-Pulido, and Juan M. Sánchez-Pérez

Univ. Extremadura. Dept. Informática
Escuela Politécnica. Campus Universitario s/n. 10071. Cáceres, Spain
mavega@unex.es, dvperez@indra.es, jangomez@unex.es, sanperez@unex.es

Abstract. Radio Network Design (RND) is a Telecommunications problem that tries to cover a certain geographical area by using the smallest number of radio antennas, and looking for the biggest cover rate. Therefore, it is an important problem, for example, in mobile/cellular technology. RND can be solved by bio-inspired algorithms, among other options, because it is an optimization problem. In this work we use the PBIL (Population-Based Incremental Learning) algorithm, that has been little studied in this field but we have obtained very good results with it. PBIL is based on genetic algorithms and competitive learning (typical in neural networks), being a new population evolution model based on probabilistic models. Due to the high number of configuration parameters of the PBIL, and because we want to test the RND problem with numerous variants, we have used grid computing with BOINC (Berkeley Open Infrastructure for Network Computing). In this way, we have been able to execute thousands of experiments in only several days using around 100 computers at the same time. In this paper we present the most interesting results from our work.

Keywords: RND, PBIL, BOINC, Evolutionary Algorithm, Antenna, Coverage, Grid Computing.

1 Introduction

Many of the problems found in the Telecommunications area can be formulated as optimization tasks. Some examples are: the frequency assignment problem (FAP), the bandwith-demand prediction in ATM networks, the error correcting code design (ECC), the design of telecommunication networks,…

The problem we study in this paper is part of the telecommunication network design problems. When a set of geographically-dispersed terminals needs to be covered by transmission antennas (also called base transceiver stations -BTS-), the key subject will be to minimize the number and localizations of those antennas and to cover the biggest possible area. This is the Radio Network Design problem (RND).

RND is an NP-hard problem, for this reason its resolution by means of bio-inspired algorithms is very appropriate. In this work we use the PBIL, a modern algorithm and

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 91–100, 2007.
© Springer-Verlag Berlin Heidelberg 2007

little used in Telecommunications problems. Although PBIL is a modern algorithm, it has been proved that it can obtain good results. It is also important to highlight that PBIL is an easy-of-parallelize algorithm.

Finally, due to the high number of parameters to configure in PBIL, as well as the great quantity of variants of the RND problem we wanted to evaluate, we have applied grid computing for the realization of our experiments. In particular, we have used BOINC, a very interesting proposal for volunteer computing and desktop grid computing.

The rest of the paper is organized as follows: section 2 briefly introduces the RND problem. After that, we detail the PBIL algorithm, and then, in section 4 we explain the fundamental aspects of BOINC. Section 5 shows the most interesting results of this work, displaying the conclusions and future work in the last section.

2 Radio Network Design

The RND problem tries to cover the biggest area with the minimal set of antennas/transmitters. Some approaches to solve this problem include the use of an evolutionary algorithm [1-2], for example, the genetic algorithms [3].

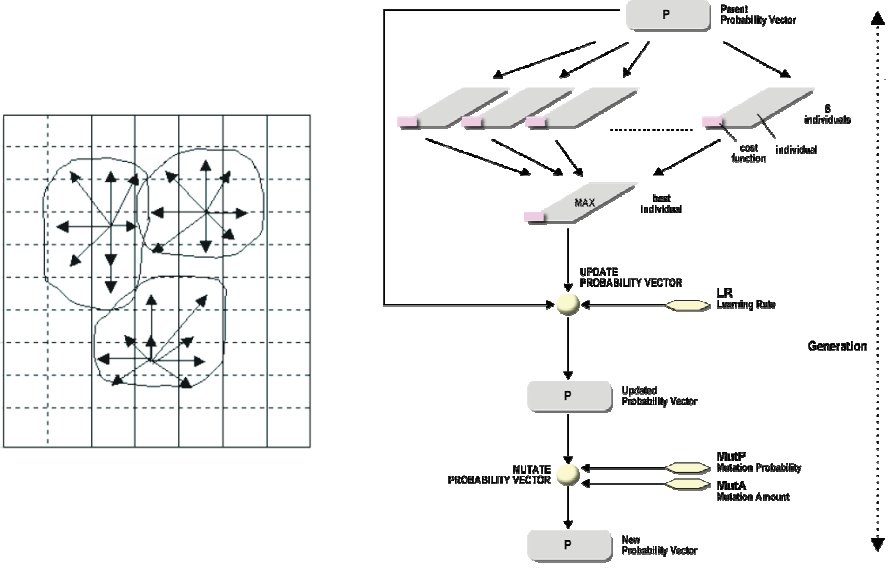


Fig. 1. (a) Three potentially transmitter locations and their associated covered cells on a grid. (b) The Population-Based Incremental Learning algorithm.

Let us consider the set L of all potentially covered locations and the set M of all potential transmitter locations. Let G be the graph, (M $\bigcup$ L, E), where E is a set of edges such that each transmitter location is linked to the locations it covers. As the geographical area needs to be discretized, the potentially covered locations are taken

from a grid, as shown in figure 1a. In our case, we focus on a 287×287 point grid representing an open-air flat area (a total of 82,369 different positions) and we will be able to use a maximum of 349 antennas. Since we apply the PBIL algorithm to solve this problem, the vector x will be a solution to the problem where $x_i \in \{0,1\}$, and $i \in [1, 349]$, indicates whether a transmitter is used (1) or not (0).

The objective of RND is to search for the minimum subset of transmitters that covers a maximum surface of an area, therefore, we are searching for a subset $M' \subseteq M$ such that $|M'|$ is minimum and such that $|Neighbours(M', E)|$ is maximum, where:

$$\begin{aligned} Neighbours(M', E) &= \{u \in L \mid \exists v \in M', (u,v) \in E\} \\ M' &= \{t \in M \mid x_t = 1\} \end{aligned} \tag{1}$$

The fitness function we have used in PBIL is shown in equation 2 [3].

$$f(x) = \frac{CoverRate(x)^2}{NumberTransmittersUsed(x)} \tag{2}$$

An important constraint in this problem consists in determining the list of available locations for the antennas, because there are some places where the antennas can not be placed (public recreation areas, etc.). In our case, the predefined set of available locations is the one obtained from [4]. This will make easy the future comparisons with other evolutionary techniques.

3 Population-Based Incremental Learning

PBIL [5-6] is a method that combines the genetic algorithms with the competitive learning (typical in artificial neural networks) for function optimization. PBIL is an extension to the EGA (Equilibrium Genetic Algorithm) achieved through the re-examination of the performance of the EGA in terms of competitive learning.

One of the fundamental attributes of the genetic algorithms is their ability to search the function space from multiple points in parallel. In this context, parallelism does not refer to the ability to parallelize the implementation of genetic algorithms; rather, it refers to the ability to represent a very large number of potential solutions in the population of a single generation. Because of the failure of this implicit parallelism to exist in the latter parts of genetic search in a genetic algorithm, the utility of maintaining multiple points, through the use of a population, decreases. The limited effectiveness of the population in the latter portions of search allows it to be modelled by a probability vector, specifying the probability of each position containing a particular value. This concept is central in PBIL.

PBIL attempts to create a probability vector from which samples can be drawn to produce the next generation's population. As in standard genetic algorithms (GAs), it is assumed that the solution is encoded into a fixed length vector. Ignoring the contribution of mutation, the expected distribution of values in each position of the population during generation G, can be calculated based upon the population of generation G-1. Assuming a fully generational change (each member of the population is replaced in the subsequent generation), fitness proportional selection, and a general pair-wise

recombination operator, the probability of value j appearing in position i in a solution vector x, in a population at generation G, can be calculated as it is shown in equation 3.

$$P(i,j) = P(x_i = j) = \frac{\sum_{v \in Population_{G-1} \wedge v_i = j} EvaluateVector(v)}{\sum_{v \in Population_{G-1}} EvaluateVector(v)} \quad (3)$$

Although we will consider the PBIL algorithm with solution vectors encoded in a binary alphabet, this algorithm can be used with other representations.

In figure 2 we can see the probability representation of 3 small populations of 4-bit solution vectors; population size is 4. In this representation, the probability vector specifies the probability of each bit position containing a '1'. It is important to highlight that the first and third representation for the population are the same, although the solution vectors (from the corresponding populations) each represents are entirely different.

Population #1	**Population #2**	**Population #3**
1010	1010	0011
0101	1100	1100
1010	1100	1100
0101	1100	0011
Representation	**Representation**	**Representation**
0.5 0.5 0.5 0.5	1.0 0.75 0.25 0.0	0.5 0.5 0.5 0.5

Fig. 2. Examples of populations and their representation using the probability vector

In conclusion, PBIL attempts to create a probability vector which can be considered a prototype for high-evaluation vectors (after the fitness function) for the function space being explored. The following points describe the PBIL algorithm performance (figure 1b). PBIL is an iterative algorithm, where the successive generations try to get an optimal solution, stopping when the maximum number of generations is reached or when the fitness of the current solution is greater than a predetermined value:

1. Establish all the necessary parameters for PBIL: the population size (S, number of samples/individuals to produce per generation), the probability of mutation occurring in each position of the probability vector (MutP), the amount for mutation to affect the probability vector (MutA) and the learning rate (LR).
2. Initialize probability vector P (each position = 0.5).
3. Generate the S samples (individuals in the population). Each sample vector must be generated according to probabilities in P. Furthermore, each sample vector is also evaluated using the fitness function.
4. Find the best sample MAX, that is, find the sample vector corresponding to maximum evaluation (after the fitness function).

5. Update the probability vector P, position by position, using the sample MAX and the learning rate LR: Pi ← Pi * (1.0 - LR) + MAXi * (LR).
6. Mutate the probability vector P, position by position, using the mutation probability MutP and the mutation amount MutA; and in such a way that for each position of the probability vector P we evaluate if a random number (between 0 and 1, range of probabilities) is less than MutP. In this case, and only in this case, we apply the equation: Pi ← Pi * (1.0 – MutA) + random (0.0 or 1.0) * (MutA).
7. Then, the next generation begins in the above third point.

4 Grid Computing with BOINC

At present we are using the middleware system BOINC [7-8] (Berkeley Open Infrastructure for Network Computing) in order to perform massive computations/experiments in a parallel way for PBIL. BOINC is a system for "volunteer computing" and "desktop grid computing". Volunteer computing uses computers volunteered by the general public to do distributed scientific computing.

In our case, we use BOINC in order to perform many different executions of our evolutionary algorithm in parallel. In this way, we can do a deep survey about which are the best parameters and combinations for solving the RND problem. People interested in learning more about our platform RND-BOINC (RND@home), or that want to join in this project (volunteer donation of CPU cycles), can access it via the website http://arcoboinc.unex.es/rnd. At present, around 100 computers are joining in this project, executing hundred of experiments at the same time (in parallel).

5 Results

In this project we have evaluated the different configuration parameters for our PBIL algorithm in order to solve the RND problem. The total number of possible combinations is very high, for space reasons, in this paper we only show some of the most interesting experiments.

In particular, in this project we have planned a total of 25280 different experiments. Within those 25280 experiments, the half corresponds to antennas with square coverage and the other half to those with omnidirectional coverage. In both types of coverage we perform experiments for radius 20 and 30 (the half with each radius value). We have foreseen that a total of 32454 hours will be necessary in order to perform all these experiments. In this moment, all the square-coverage experiments have been finished (12640 experiments), and we have already done 2910 (out of 12640) for omnidirectional coverage (the computations are much slower because they need to use the circle equation, square roots, powers, etc.).

To choose a number of graphs feasible for this paper, we will do the following: we have several parameters that we can adjust in our PBIL algorithm. These parameters are: number of generations, number of samples/individuals (the population size), mutation probability, mutation amount, learning rate, negative learning rate (only valid for 1 variant of the PBIL algorithm), number of M best individuals (only valid for 3 variants of the PBIL algorithm), coverage type (square or omnidirectional), and

coverage radius size. First we have tested several values for the first parameter (number of generations) maintaining fixed the rest of them (to their value more used in the literature). Once we obtain the best value for this first parameter, we maintain this value and we vary only the following parameter (number of samples/individuals) also finding its best value, and so on, until concluding in the best adjustment for all the parameters.

In the following graphs we show in red colour the number of transmitters used (first bar), in green the fitness function (second bar), in blue the cover rate (third bar), and in yellow the average execution time (hours:minutes:seconds).

Our first experiment tests the best number of generations, keeping the rest of parameters to their value more used in the literature for other optimization problems. These values are:

- PBIL Variant = Standard
- Number of individuals = 100
- Learning rate = 0.10
- Mutation probability = 0.02
- Mutation Amount = 0.05
- Coverage type = Square
- Radius size = 20 (each transmitter has an associated 41×41 point cell)

We study the following values for the number of generations: 1000, 2500, 5000, 7500 and 10000. Figure 3a shows the results. We can see that we obtain the best fitness function with 7500 and 10000 generations. In this case, we select 7500 as the optimal value for the number of generations because it takes less execution time. In this graph, we can also observe an almost evident conclusion: the more generations, the better results (but more execution time).

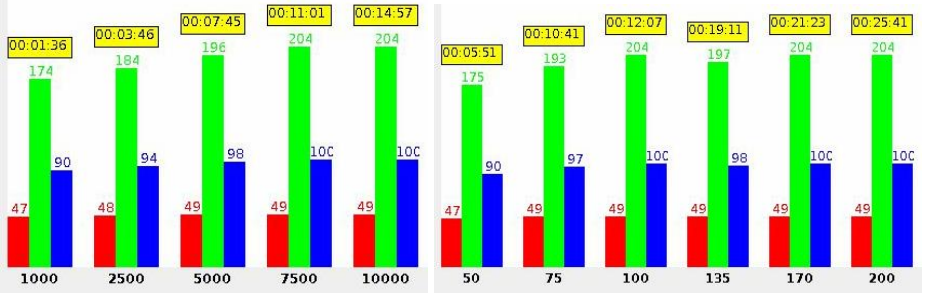


Fig. 3. (a) Number of generations. (b) Number of samples/individuals.

The next experiment tries to select the best value for the number of samples (population size), keeping the previous values for the rest of parameters except for the number of generations, whose value will be 7500, our best result in the previous experiment. Figure 3b indicates that the best number of individuals is 100 (it obtains the best possible result, fitness = 204.082, in less time).

Our following test searches for the best learning rate (LR), keeping the rest of parameters with their initial value, except for generations = 7500 and individuals =

100. In this case, the best result is obtained with LR = 0.10 (figure 4a). It is also important to highlight that the value 0.10 is the inflection point: the more far from this value, the worse fitness result.

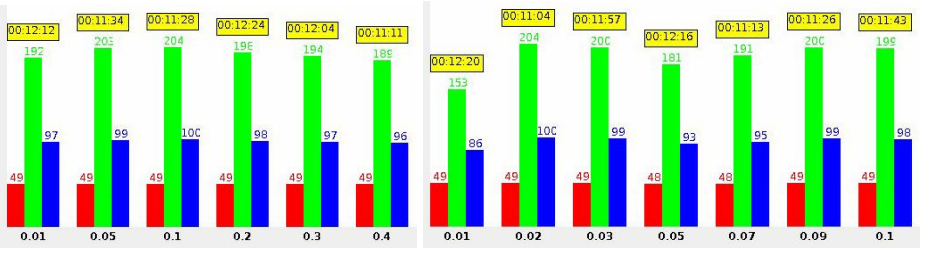


Fig. 4. (a) Learning rate. (b) Mutation probability.

Now, we test the mutation probability (as always, we keep the rest of parameters with the best values obtained in the previous experiments). As figure 4b displays, among the values 0.01, 0.02, 0.03, 0.05, 0.07, 0.09 and 0.10, the best value is 0.02. As we have done for other parameters, we will use this value in our next experiments.

The following experiment evaluates the mutation amount. Figure 5a shows that the best adjustment for this parameter is 0.07, because it obtains the optimal fitness value in less time. Observe that, in this point, several options reach the optimal fitness value (204.082, 100% of coverage with 49 transmitters), due to many parameters already have their optimal adjustment.

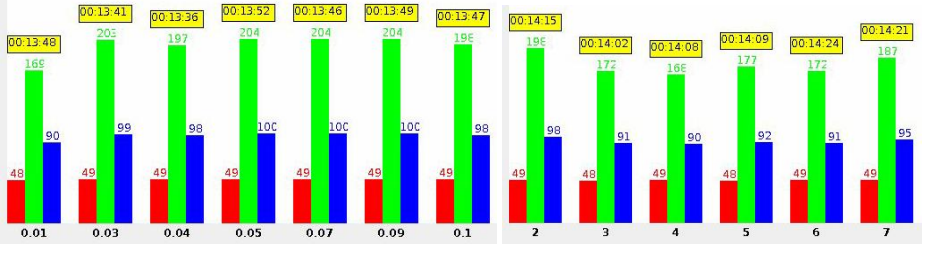


Fig. 5. (a) Mutation amount. (b) Searching for the best M in PBIL-M-Equitable.

We already have the best adjustment for all the parameters in the standard PBIL. Our next step is to evaluate other variants of PBIL (and their possible new parameters). The variants of PBIL we have studied are the following:

- PBIL-Standard.
- PBIL-Complement: The probability vector is moved towards the complement of the lowest evaluation individual (the worst sample) in each generation.
- PBIL-Different: The probability vector is only moved towards the bits in the best individual which are different than those in the worst individual, in each generation.

– PBIL-M-Equitable: The probability vector is moved equally in the direction of each of the M selected individuals (M best samples) in each generation.
– PBIL-M-Relative: The probability vector is moved on the relative evaluations (fitness functions) of the M best individuals in each generation.
– PBIL-M-Consensus: The probability vector is moved only in the positions in which there is a consensus in all of the M best individuals in each generation.
– PBIL-NegativeLR: The probability vector is moved towards the best individual (using the learning rate) and also away from the worst individual (using the negative learning rate) in each generation.

The next 3 experiments search for the best M in the variants PBIL-M-Equitable, PBIL-M-Relative and PBIL-M-Consensus. The experiments are shown, respectively, in figures 5a, 6a and 6b. In all the experiments the best value is M=2. We highlight the experiment with PBIL-M-Consensus, where all the M values obtain very similar results. In this case, we select M=2 in order to use the same M in all the PBIL variants.

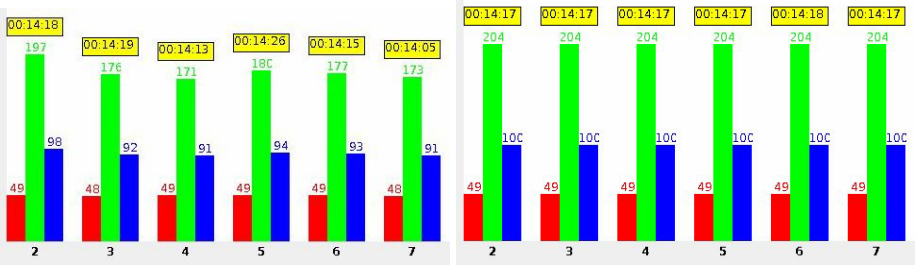


Fig. 6. (a) The best M in PBIL-M-Relative. (b) The best M in PBIL-M-Consensus.

In the following experiment we evaluate the best value for the negative learning rate in the PBIL-NegativeLR (figure 7a). Many values obtain optimal results, but we select the value 0.07 because takes less time.

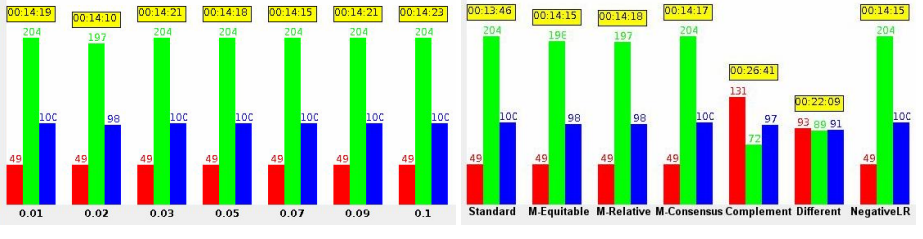


Fig. 7. (a) Negative learning rate. (b) Searching for the best PBIL variant.

Our final experiment tests all the PBIL variants (with their best adjustments) in order to know which of them is the best option. As we can see in figure 7b, the best option is the standard PBIL. The PBIL-NegativeLR and the PBIL-M-Consensus also obtain very good results but they take more time. Furthermore, it is important to

highlight that the PBIL-Complement and PBIL-Different obtained very bad results for the RND problem, although these variants are also proposed in the literature [5]. Table 1 summarizes the results presented in the previous experiments. Observe that the standard PBIL does not use the parameters negative learning rate and M.

Table 1. The best result with PBIL for solving the RND problem with square BTS (radius 20)

	Configuration of the best result
PBIL variant	Standard
# Generations	7500
# Individuals	100
Mutation probability	0.02
Mutation amount	0.07
Learning rate	0.10
Negative learning rate	0.07 - Not used
M best individuals	2 - Not used

	Results
Fitness function	204.082
# Transmitters	49
Coverage	100%
Execution time	13 minutes & 46 seconds
Execution on	Pentium IV – 2.8 GHz
# Evaluations	712,200

5 Conclusions

In this paper we have presented a deep analysis about the use of PBIL, a modern evolutionary algorithm little used in telecommunication problems, for solving the RND problem, an important network design problem (for example, in mobile/cellular technology). In order to perform this deep analysis we have carried out thousands of experiments. For this reason, and in order to speedup these computations, we have used grid computing with BOINC, performing many different executions of our evolutionary algorithm in parallel (currently, in around 100 computers).

We think we have obtained very good results during this deep survey, as we have presented in the previous section. In particular, table 2 displays a comparison about the results included in this paper and the results presented by other authors in the literature (also for square BTS and radius 20, that is, each transmitter has an associated 41×41 point cell).

Table 2. Comparison between our best result and other results in the literature

Evolutionary technique	Fitness function	Execution time	Execution on	Reference
PBIL (Population-Based Incremental Learning)	204.082	13 minutes & 46 seconds	1 Pentium IV 2.8 GHz	This work
SGA (Standard Genetic Algorithm)	125.4	9 hours	1 Sparc-4 workstation	[1]
IPGA (Island-based Parallel Genetic Algorithm)	193.8	17 minutes	40 Sparc-4 workstations connected by FDDI and Ethernet	[1]
ssGA (steady state Genetic Algorithm)	204.082	3.86 hours	1 Ultra Sparc-1 143 MHz	[3]
dssGA (distributed ssGA)	204.082	10.87 hours	1 Ultra Sparc-1 143 MHz	[3]
dssGA (distributed ssGA)	204.082	1.38 hour	8 Ultra Sparc-1 143 MHz connected by ATM network	[3]

In this table, the SGA has two main problems: (a) it is rather slow, and therefore it can not be used interactively as required by telecommunication operators; (b) the solutions obtained remain far from the optimal solution. The IPGA gets reasonable solutions in acceptable time, but it needs 40 Sparc-4 workstations working in parallel. The rest of evolutionary techniques obtain the optimal solution to this RND problem (fitness value = 204.082). As we can see, the PBIL outperforms other existing parallel and sequential evolutionary approaches. Since the actual goal of a telecommunication network designer is to get an optimum design at a maximum speed, we think the PBIL is the best suited algorithm to this task. This is partially due, of course, to the fact that we are using different machines, but observe that some proposals even use NOW (networks of workstations) and parallel evolutionary algorithms, and we have obtained these results executing our sequential version of PBIL in an only PC. On the other hand, we need to remark that the PBIL solves a more difficult instance problem due to all the other references only consider a total number of 149 possible transmitter sites, while the PBIL considers a total of 349 possible sites (the size of our problem instance is clearly higher, and therefore, more difficult to solve and time-consuming).

Future work includes the study of other bio-inspired algorithms, as the Differential Evolution [9], searching for new results and comparisons. In this line, we will also use grid computing with BOINC in order to speedup all our experiments. Other future line is the solution to the RND problem considering BTS locations from real city maps.

Acknowledgments. This work has been partially funded by the Ministry of Education and Science and FEDER under contract TIN2005-08818-C04-03 (project OPLINK).

References

1. P. Calégari, F. Guidec, P. Kuonen, D. Kobler. Parallel Island-Based Genetic Algorithm for Radio Network Design. Journal of Parallel and Distributed Computing, 47(1): 86-90, November 1997.
2. P. Calégari, F. Guidec, P. Kuonen, F. Nielsen. Combinatorial Optimization Algorithms for Radio Network Planning. Theoretical Computer Science, 263(1): 235-265, July 2001.
3. E. Alba. Evolutionary Algorithms for Optimal Placement of Antennae in Radio Network Design. NIDISC'2004 Sixth International Workshop on Nature Inspired Distributed Computing, IEEE IPDPS, Santa Fe, USA, pp. 168-175, April 2004.
4. OPLINK: http://oplink.lcc.uma.es/problems/rnd.html, 2006.
5. S. Baluja. Population-based Incremental Learning: A Method for Integrating Genetic Search based Function Optimization and Competitive Learning. Technical Report CMU-CS-94-163, Carnegie Mellon University, June 1994.
6. S. Baluja, R. Caruana. Removing the Genetics from the Standard Genetic Algorithm. Twelfth International Conference on Machine Learning, San Mateo, CA, USA, pp. 38-46, May 1995.
7. BOINC: http://boinc.berkeley.edu, 2006.
8. D.P. Anderson. BOINC: A System for Public-Resource Computing and Storage. 5th IEEE/ACM International Workshop on Grid Computing, Pittsburgh, USA, pp. 365-372, November 2004.
9. K. Price, R. Storn. Differential Evolution – A Simple Evolution Strategy for Fast Optimization. Dr. Dobb's Journal, 22(4): 18–24, April 1997.

Evaluation of Different Metaheuristics Solving the RND Problem

Miguel A. Vega-Rodríguez[1], Juan A. Gómez-Pulido[1], Enrique Alba[2], David Vega-Pérez[1], Silvio Priem-Mendes[3], and Guillermo Molina[2]

[1] Dep. of Computer Science, Univ. of Extremadura, Caceres, Spain
{mavega,jangomez}@unex.es, dvperez@indra.es
[2] Dep. of Computer Science, Univ. of Malaga, Malaga, Spain
{eat,guillermo}@lcc.uma.es
[3] Polytechnic Institute of Leiria, High School of Technology, Leiria, Portugal
smendes@estg.ipleiria.pt

Abstract. RND (Radio Network Design) is a Telecommunication problem consisting in covering a certain geographical area by using the smallest number of radio antennas achieving the biggest cover rate. This is an important problem, for example, in mobile/cellular technology. RND can be solved by bio-inspired algorithms. In this work we use different metaheuristics to tackle this problem. PBIL (Population-Based Incremental Learning), based on genetic algorithms and competitive learning (typical in neural networks), is a population evolution model based on probabilistic models. DE (Differential Evolution) is a very simple population-based stochastic function minimizer used in a wide range of optimization problems, including multi-objective optimization. SA (Simulated Annealing) is a classic trajectory descent optimization technique. CHC is a particular class of evolutionary algorithm which does not use mutation and relies instead on incest prevention and disruptive crossover. Due to the complexity of such a large analysis including so many techniques, we have used not only sequential algorithms, but grid computing with BOINC in order to execute thousands of experiments in only several days using around 100 computers. In this paper we present the most interesting results from our work, indicating the pros and cons of the studied solvers.

Keywords: RND, PBIL, DE, SA, CHC, Metaheuristics, Evolutionary Techniques.

1 Introduction

The Radio Network Design problem is a kind of telecommunication network design problem. When a set of geographically-dispersed terminals needs to be covered by transmission antennas (also called base station transmitters or base transceiver stations -BTS-), a capital subject is to minimize the number and locations of those antennas and to cover the largest area.

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 101–110, 2007.
© Springer-Verlag Berlin Heidelberg 2007

RND is an NP-hard problem; therefore its solution by means of bio-inspired algorithms is appropriate. In this work we use several different metaheuristics in order to solve this problem: PBIL, DE, SA and CHC.

Finally, since our interest is not only studying these techniques, but also to open new research lines, we have applied grid computing for the realization of our experiments (not in an exhaustive manner for space constraints in this conference paper). In particular, we have used BOINC (Berkeley Open Infrastructure for Network Computing), a very interesting proposal for volunteer computing and desktop grid computing.

The rest of the paper is organized as follows: Section 2 briefly explains the RND problem. After that, in the following sections we introduce the PBIL, DE, SA and CHC algorithms. Then, in Section 7 we show the most interesting results of this work, including comparisons among the different studied techniques, finally leading to the conclusions and future work summarized in the last section.

2 The RND Problem

The RND problem [1,2] consists in covering a largest area with a minimal set of transmitters. In order to mathematically define this problem, let us consider the set L of all potentially covered locations and the set M of all potential transmitter locations. Let G be the graph, (M $\bigcup$ L, E), where E is a set of edges such that each transmitter location is linked to the locations it covers. As the geographical area needs to be discretized, the potentially covered locations are taken from a grid, as shown in Figure 1a. In our case, we focus on a 287×287 point grid representing an open-air flat area (a total of 82,369 different positions) and we will be able to use a maximum of 349 available locations for placing antennas. The vector **x** will be a solution to the problem where $x_i \in \{0,1\}$, and $i \in [1, 349]$, indicates whether a transmitter is used (1) or not (0) in the corresponding location.

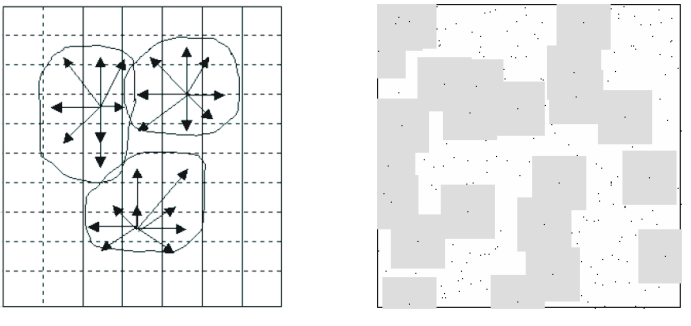


Fig. 1. (a) Three potential transmitter locations and their associated covered cells on a grid. (b) Base station transmitters with square coverage.

The objective of RND is to search for the minimum subset of transmitters that covers a maximum surface of an area, therefore, we are searching for a subset M' $\subseteq$ M such that |M'| is minimum and such that |Neighbours(M', E)| is maximum,

where:

$$\text{Neighbours}(M', E) = \{u \in L \mid \exists v \in M', (u, v) \in E\}$$
$$M' = \{t \in M \mid x_t = 1\} \quad (1)$$

The fitness function we have used in our experiments is shown in Equation 2 [3].

$$f(x) = \frac{CoverRate(x)^2}{NumberTransmittersUsed(x)} \quad (2)$$

An important constraint in this problem consists in determining the list of available locations for the antennas, because there are some places where the antennas can not be placed (public recreation areas, etc.). In our case, for the predefined set of available locations we selected the one included in [4]. This will make easy the comparisons among the different evolutionary techniques.

In our experiments we consider base station transmitters with square cells (see figure 1b), each transmitter has an associated 41×41 point cell (coverage radius size is 20). Other cell shapes are possible (like circular cells for omnidirectional antennas) but deferred for future work. Let us now briefly present the used algorithms.

3 Population-Based Incremental Learning

Population-Based Incremental Learning (PBIL) is a method that combines a genetic algorithm with competitive learning for function optimization. Instead of applying operators, PBIL infers a probability distribution from the present population and samples the new population from the inferred distribution [5,6].

4 Differential Evolution

Differential Evolution (DE) is an algorithm used for continuous optimization problems in the past with satisfactory results [7,8]. DE is a simple population-based stochastic function minimizer/maximizer, used in a wide range of optimization problems, including multi-objective optimization [9]. It has been modified in this research to work with discrete representations [10].

5 Simulated Annealing

Simulated annealing (SA) is a generic probabilistic meta-algorithm for the global optimization problem, namely locating a good approximation to the global optimum of a given function in a large search space. It was independently invented by S. Kirkpatrick, C. D. Gelatt and M. P. Vecchi in 1983 [11], and by V. Černý in 1985 [12]. SA is a trajectory based optimization technique (i.e., only one tentative solution is manipulated in contrast with the rest of algorithms here, where a population of solutions is used). It is commonly found in industry and provides good results; therefore it constitutes an interesting method for comparison.

6 CHC

The fourth algorithm we propose for solving the RND problem is Eshelman's CHC [13], which stands for Cross-generational elitist selection, Heterogeneous recombination (by incest prevention), and Cataclysmic mutation. CHC is a kind of Evolutionary Algorithm (EA), where mutation is not used. As a mechanism for preventing convergence and maintaining diversity, CHC employs incest prevention [14] plus a special recombination procedure known as HUX, combined with a special restarting mechanism for adding diversity through mutation when stagnation is detected (cataclysmic mutation).

7 Results

In this section we present the most interesting results coming from using each of the evolutionary techniques we have proposed to solve the RND problem.

7.1 Results Using PBIL

In this work we have first evaluated different configuration parameters for our PBIL algorithm in order to solve the RND problem since this application to this problem is relatively new. In particular, the parameters we can adjust in PBIL are: used variant of PBIL (we have 7 variants explained in the next paragraphs), number of samples (the population size), mutation probability, mutation shift (intensity of the mutation that affects the probability vector), learning rate, negative learning rate (only valid for the variant PBIL-NegativeLR), number of M best individuals (only valid for the variants PBIL-M-Equitable, PBIL-M-Relative and PBIL-M-Consensus), and whether an elitist strategy is used or not (the best individual in the previous population is transferred to the current generation unaltered).

The variants of PBIL we have thus studied are the following:

- PBIL-Standard.
- PBIL-Complement: The probability vector is moved towards the complement of the lowest evaluation individual (the worst sample) in each generation.
- PBIL-Different: The probability vector is only moved towards the bits in the best individual which are different than those in the worst individual, in each generation.
- PBIL-M-Equitable: The probability vector is moved equally in the direction of each of the M selected individuals (M best samples) in each generation.
- PBIL-M-Relative: The probability vector is moved on the relative evaluations (fitness functions) of the M best individuals in each generation.
- PBIL-M-Consensus: The probability vector is moved only in the positions in which there is a consensus (the same value) in all of the M best individuals in each generation.
- PBIL-NegativeLR: The probability vector is moved towards the best individual (using the learning rate) and also away from the worst individual (using the negative learning rate) in each generation.

For the rest of configuration parameters we present a wide analysis including the following values (most typical in literature):

- Number of generations: 1000, 2500, 5000, 7500 and 10000.
- Number of samples (the population size): 50, 75, 100, 135, 170 and 200.
- Mutation probability: 0.01, 0.02, 0.03, 0.05, 0.07, 0.09 and 0.10.
- Mutation shift: 0.01, 0.03, 0.04, 0.05, 0.07, 0.09 and 0.10.
- Learning rate:0.01, 0.05, 0.10, 0.20, 0.30 and 0.40
- Negative learning rate: 0.01, 0.02, 0.03, 0.05, 0.07, 0.09 and 0.10.
- Number of M best individuals: 2, 3, 4, 5, 6 and 7.
- Elitism: Yes or no.

The total number of possible combinations is very high, for this reason we have used the middleware system BOINC [15,16] (Berkeley Open Infrastructure for Network Computing) in order to perform massive computations/experiments in a parallel way for PBIL. BOINC is a system for volunteer computing and desktop grid computing. Volunteer computing uses computers volunteered by the general public to do distributed scientific computing.

In our case, we use BOINC in order to perform many different executions of our evolutionary algorithm in parallel. In this way, we can do a deep survey about which are the best parameter values for solving the RND problem with PBIL, which is needed to guide future research after this first study. Researchers interested in learning more about our platform RND-BOINC (RND@home), or wanting to join in this project (volunteer donation of CPU cycles), can access it via the website http://arcoboinc.unex.es/rnd. At present, around 100 computers are available in this project, executing hundreds of experiments at the same time in parallel.

Table 1. The most important results with PBIL for solving the RND problem

	Config. best result
PBIL variant	Standard
# Generations	2500
# Individuals	135
Mutation probability	0.05
Mutation shift	0.07
Learning rate	0.30
Negative learning rate	---
M best individuals	---
Elitism	YES

	Best result	Average results
Fitness function	204.082	204.082
# Transmitters	49	49
Coverage	100%	100%
Execution time	5 minutes, 34 seconds	5 minutes, 56 seconds
Execution on	Pentium IV – 2.8 GHz	
Normalized execution time (P.IV-1.7 GHz)	9 minutes, 10 seconds	9 minutes, 46 seconds
# Evaluations	276,345	306,855

Table 1 shows the most important results using PBIL. In particular, for every combination of parameters, 30 independent runs have been performed for statistical purposes. In this table, the normalized execution time converts the obtained execution time to a virtual standard Pentium IV-1.7 GHz. As we can see, PBIL solved the problem with 100% accuracy, and presents low variability between the best computational effort (276,345 evaluations) and the average (306,855 evaluations). Furthermore, we can observe PBIL is quite fast.

7.2 Results Using DE

In order to compare the results we have obtained from DE with other optimization methods and techniques, it is necessary to apply the experiments on the same predefined set of BS available locations.

We have programmed two types of experiments:

- In the first type, we try to find the optimal set of locations (reaching the 100% of coverage) for a fixed amount of BS transmitters (49). The parameters of DE to be initially set are: population size, crossover function and maximum number of generations to be performed.
- The second type of experiments consists in looking for the minimum number of transmitters to be located in the target area in order to reach a predetermined cover rate. A loop goes on starting from an initial amount of transmitters and increasing it until the wanted coverage is obtained. For every number of BS in this loop, DE is applied in the same manner as in the first type of experiments (see above).

Two crossover functions have been considered: FA and SA. Let be two set of locations (individuals), named A and B (the parents). Let be the S individual (the offspring) obtained from the application of the crossover function to A and B. FA function chooses the first half of A to build the first half of the offspring. The second half of the offspring is then built with the first half of B, but if a repeated location appears, successive locations of the second halves of B and A are taken. SA function chooses the second half of A to build the first half of the offspring, and the second half of the offspring is then built with the second half of B, but if a repeated location appears, successive locations of the first halves of B and A are taken.

With this background we have performed a series of tests for both types of experiments. For every combination of parameters, 30 independent runs have been performed for statistical purposes. We have observed than FA crossover produces better results than SA crossover, and furthermore, these results are obtained in a lower number of generations (conclusions obtained mainly from the second type of experiment). Table 2 shows the most important results for the first type of experiment, considering the same instance of the problem used by the other algorithms presented in this paper. The main conclusion is that the desired optimal coverage (100%) has not been reached. Perhaps this result could be improved with more evaluations. Also, you can observe DE algorithm is fast.

Table 2. The most important results with DE for solving the RND problem

	Config. best result
# Generations	4000
# Individuals	2000
Crossover	FA

	Best result	Average results
Fitness function	163.48	163.48
# Transmitters	49	49
Coverage	89.5%	89.5%
Execution time	3 minutes	3 minutes, 48 seconds
Execution on	Pentium IV –1.7 GHz	
# Evaluations	9,363	11,538

7.3 Results Using SA

SA has been used on the same instance as the previous algorithms, in order for the obtained results to be comparable.

SA has only three parameters the programmer needs to tune:

- The mutation probability. The values employed range from 0.005 to 0.97.
- The length of the Markov chain.
- The temperature decay, α. The values employed range from 0.99 to 0.99999.

The length of the Markov chain and the temperature decay have been proven to work in the same manner, thus to be equivalent. Therefore, we decided to keep the first at a constant value of 50, and allow the tuning of the latter.

For every combination of parameters, 50 independent runs have been performed in order to assure its statistical relevance. Table 3 shows the results. The tests have been performed in a 16 machine cluster named in dedicated mode, and the code has been developed using the MALLBA library [17]. This resource code is available at the web page http://neo.lcc.uma.es/mallba/easy-mallba/index.html.

SA has been able to solve the problem with 100% accuracy, but presents a high degree of variability between the best computational effort (441,141 evaluations) and the average (810,755 evaluations).

Table 3. The most important results with SA for solving the RND problem

	Config. best result
# Evaluations	5,000,000
Mutation probability	0.005
Markov chain length	50
Temperature decay	0.999
Initial temperature	1

	Best result	Average results
Fitness function	204.082	204.082
# Transmitters	49	49
Coverage	100%	100%
Execution time	7 minutes	12 minutes, 10 seconds
Execution on	Pentium IV – 2.4 GHz	
Normalized execution time (P.IV-1.7 GHz)	9 minutes, 53 seconds	17 minutes, 11 seconds
# Evaluations	441,141	810,755

7.4 Results Using CHC

When using the CHC algorithm on the same instance that the previous methods, we have considered two parameters that can be tuned:

- The population size. The values employed range from 50 to 10,000 individuals.
- The cataclysmic mutation probability. Ranging from 0.01 to 0.35.

50 independent runs are performed for every combination of parameters. Table 4 shows the best configuration, and the results obtained. The tests have been performed in a 16 machine cluster named in dedicated mode, and the code has been developed using the MALLBA library [17]. This resource code is available at the web page http://neo.lcc.uma.es/mallba/easy-mallba/index.html.

During the tests we have concluded that the mutation probability (second parameter tuned) has little effect on the algorithm's performance, and can be kept at a value of 35% without any significant loss of efficiency.

CHC solved the problem with 100% accuracy, and presents low variability in the computational effort: there is little difference between the best effort (291,200 evaluations) and the average (380,183 evaluations).

Table 4. The most important results with CHC for solving the RND problem

	Config. best result
# Evaluations	5,000,000
# Individuals	2,800
Mutation probability	0.35
Incest distance	25% vector length
Crossover probability	0.8
Elitism	YES

	Best result	Average results
Fitness function	204.082	204.082
# Transmitters	49	49
Coverage	100%	100%
Execution time	5 minutes, 58 seconds	7 minutes, 8 seconds
Execution on	Pentium IV – 2.4 GHz	
Normalized execution time (P.IV-1.7 GHz)	8 minutes, 25 seconds	10 minutes, 4 seconds
# Evaluations	291,200	380,183

8 Conclusions

In this paper we have solved the RND (Radio Network Design) problem with different metaheuristic techniques. Our aim was to solve the problem efficiently and at the same time research in the results of a wide spectrum of modern techniques. Most of these techniques (PBIL, SA and CHC) have obtained the optimal solution for this problem with square BTS of radius 20 (100% of coverage is attained placing 49 antennas, getting a fitness value of 204.082) except for the DE. However, DE is the evolutionary approach that needs lower times and number of evaluations in order to obtain a reasonable result. The difficulties in DE's accuracy may be originated by its intrinsic continuous nature.

If we look for the optimal solution, the best normalized execution times (around 9 minutes) and the best number of evaluations (below 300,000) is simultaneously attained by PBIL and CHC. Figure 2 shows the best result for each technique, both in normalized execution time (left) and in number of evaluations (right).

Figure 3 presents the same data but using the average results for each technique. In this case, the results are very similar to the previous ones: PBIL and CHC obtain a similar normalized execution time (around 10 minutes), but PBIL needs slightly less evaluations (306,855 vs. 380,183 -statistically similar). On the other hand, SA also reaches the optimal fitness value (204.082) but needs huge computational resources (time and evaluations) to obtain that result.

Future work includes the study of other bio-inspired algorithms, such as genetic algorithms, parallel genetic algorithms... in our quest for more efficient and accurate solvers for this problem. In this line, we will continue using grid computing with

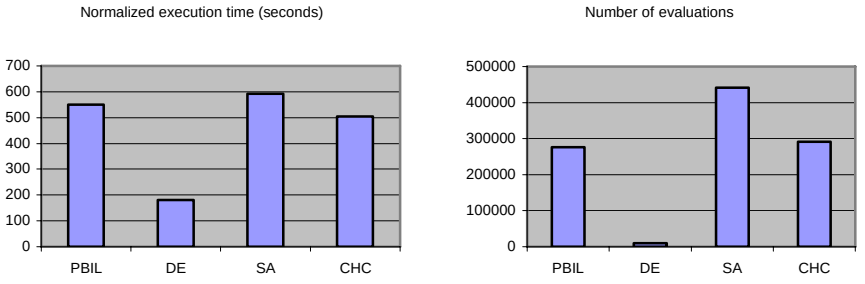


Fig. 2. Best result for each algorithm: (left) Normalized execution time (supposing a Pentium IV-1.7 GHz). (right) Number of evaluations.

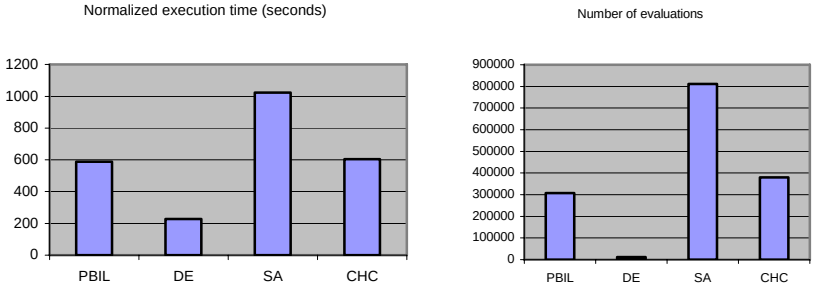


Fig. 3. Average results for each algorithm: (left) Normalized execution time (supposing a Pentium IV-1.7 GHz). (right) Number of evaluations.

BOINC and cluster computing in order to speedup all our experiments and to create more sophisticated algorithms by communicating information among component parallel agents.

Acknowledgments. This work has been partially funded by the Ministry of Education and Science and FEDER under contract TIN2005-08818-C04-01 and TIN2005-08818-C04-03 (the OPLINK project). Guillermo Molina is supported by grant AP2005-0914 from the Spanish government.

References

1. P. Calégari, F. Guidec, P. Kuonen, D. Kobler. Parallel Island-Based Genetic Algorithm for Radio Network Design. Journal of Parallel and Distributed Computing, 47(1): 86-90, November 1997.
2. P. Calégari, F. Guidec, P. Kuonen, F. Nielsen. Combinatorial Optimization Algorithms for Radio Network Planning. Theoretical Computer Science, 263(1): 235-265, July 2001.

3. E. Alba. Evolutionary Algorithms for Optimal Placement of Antennae in Radio Network Design. NIDISC'2004 Sixth International Workshop on Nature Inspired Distributed Computing, IEEE IPDPS, Santa Fe, USA, pp. 168-175, April 2004.
4. OPLINK: http://oplink.lcc.uma.es/problems/rnd.html, November 2006.
5. S. Baluja. Population-based Incremental Learning: A Method for Integrating Genetic Search based Function Optimization and Competitive Learning. Technical Report CMU-CS-94-163, Carnegie Mellon University, June 1994.
6. S. Baluja, R. Caruana. Removing the Genetics from the Standard Genetic Algorithm. Twelfth International Conference on Machine Learning, San Mateo, CA, USA, pp. 38-46, May 1995.
7. K. Price, R. Storn. Differential Evolution – A Simple Evolution Strategy for Fast Optimization. Dr. Dobb's Journal, 22(4): 18–24, April 1997.
8. K. Price, R. Storn. Web site of DE. http://www.ICSI.Berkeley.edu/~storn/code.html, November 2006.
9. H.A. Abbass, R. Sarker. The Pareto Differential Evolution Algorithm. Int. Journal on Artificial Intelligence Tools, 11(4): 531-552, 2002.
10. S. Mendes, J.A. Gómez, M.A. Vega, J.M. Sánchez. The Optimal Number and Locations of Base Station Transmitters in a Radio Network. 3rd Int. Workshop on Mathematical Techniques and Problems in Telecommunications, Leiria, Portugal, pp.17-20, September 2006.
11. S. Kirkpatrick, C.D. Gelatt, M.P. Vecchi. Optimization by Simulated Annealing. Science, 220(4598): 671-680, May 1983.
12. V. Cerny. A Thermodynamical Approach to the Travelling Salesman Problem: an Efficient Simulation Algorithm. Journal of Optimization Theory and Applications, 45: 41-51, 1985.
13. L.J. Eshelman. The CHC Adaptive Search Algorithm: How to Have Safe Search when Engaging in Nontraditional Genetic Recombination. Foundations of Genetic Algorithms, Morgan Kaufmann, pp. 265-283, 1991.
14. L.J. Eshelman, J.D. Schaffer. Preventing Premature Convergence in Genetic Algorithms by Preventing Incest. Fourth Int. Conf. on Genetic Algorithms, San Mateo, CA, USA, pp. 115-122, 1991.
15. BOINC: http://boinc.berkeley.edu, November 2006.
16. D.P. Anderson. BOINC: A System for Public-Resource Computing and Storage. 5th IEEE/ACM International Workshop on Grid Computing, Pittsburgh, USA, pp. 365-372, November 2004.
17. E. Alba, F. Almeida, M. Blesa, C. Cotta, M. Díaz, I. Dorta, J. Gabarró, C. León, G. Luque, J. Petit, C. Rodríguez, A. Rojas, F. Xhafa. Efficient Parallel LAN/WAN Algorithms for Optimization: The MALLBA Project. Parallel Computing, 32 (5-6): 415-440, June 2006.

A Comparative Investigation on Heuristic Optimization of WCDMA Radio Networks

Mehmet E. Aydin*, Jun Yang, and Jie Zhang

Center for Wireless Network Design (CWIND),
Dept. of Computing and Information Systems, University of Bedfordshire, UK
{mehmet.aydin,jun.yang,jie.zhang}@beds.ac.uk

Abstract. The planning and optimization of WCDMA (wideband code-division multiple access)radio network issues remain vital, and are carried out using static snapshot-based simulation. To improve the accuracy of the static simulation, link-level performance factors, such as the impact of power control, pilot power and soft handover, have to be taken into account. These factors have not been investigated together in the previous works. In this paper, we give a brief introduction to our programming models that take these characteristics into account, and present optimisation strategies based on three major Metaheuristics; genetic algorithms (GA), simulated annealing (SA) and tabu search (TS).. Extensive experimental results are provided and the performance of different heuristic algorithms is compared.

1 Introduction

WCDMA networks as highly inter-dependent and interference-limited systems, offer a sophisticated air interface. However, planning and optimization of WCDMA radio networks is often carried out with the static snapshot simulations for the sake of simplicity and time limitation, therefore, importing the link-level performance has been a trend to improve the performance of network planning and optimization [1]. The capacity and coverage of a WCDMA network are tightly coupled, where the coverage is generally uplink (UL) limited and the capacity can be either uplink or downlink (DL) limited depending on the uplink loading, and the base station (BS) transmission power, etc. Consequently, the performance of both UL and DL should be extensively investigated [1]. Power control (PC) and handover control are two important radio resource management (RRM) mechanisms of WCDMA, and soft handover (SHO) is a feature specific to CDMA system. Since both PC and SHO mechanisms have significant impacts on the network performance, their behaviours should be modelled accurately to obtain sensible and reliable results of network planning and optimization.

Cellular network planning and optimization is not a new topic, but as new technologies emerge, the subject remains as fresh as before. It has been proven that WCDMA radio network optimization is a NP-hard problem as computational time

* Corresponding author.

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 111–120, 2007.
© Springer-Verlag Berlin Heidelberg 2007

taken grows in a non-polynomial way with the increase of the problem sizes [2], therefore, meta-heuristics rather than other exact optimization methods are more suitable. Discrete optimization models based on BS location problem are proposed in [2] and [3], considering the impact of traffic distribution, signal quality requirements and PC mechanism. Greedy search, tabu search (TS) [2] and classical combinatorial optimisation [3] algorithms are developed to solve theses problems. An automatic two-phase network planning approach based on successively solved instances of integer programming model is discussed and the models are solved with heuristic algorithm in [4]. However, the impacts of SHO, the pilot power (by [2, 4]) and the impact of power control and the interference-limited behaviour (by [3]) are neglected. In [5], the derivative of the uplink network capacity is calculated with respect to pilot power, BS locations, and power compensation factors. These derivatives are then used in an optimization procedure to maximize the network capacity. The problem of making decision on the location and capacity of each new BS to cover expanded and increased traffic demand is studied in [6]. The objective is to minimize the cost of new BSs with a TS algorithm. Since the research is mainly based on general cellular networks, and the paper only addresses the problem of radio network expansion, the behaviour of WCDMA network is not modeled accurately. In [7], an optimization framework based on simulated annealing (SA) is presented, considering site selection and BS configuration. The framework can be used to generate completely new networks or to augment preexisting networks. However, the research mainly focuses on the general cellular networks. Likewise, [8] and [9] present SA approaches with various neighbourhood structures to solve correspondingly modelled WSDMA network problems. On the other hand, genetic algorithm (GA) based heuristic algorithms [10-12] are implemented to solve location and antenna problems in single and multiple objective optimisation point of view.

In this paper, we report a comparative study on heuristic optimisation of WCDMA networks considering CPICH power and SHO in the model. We have not come across any study consider all these factors together for optimisation with the metaheuristic algorithms examined. The remainder of this paper is organized as follows. A brief introduction to the WCDMA network planning problems with consideration to the factors mentioned is presented in Section 2. Optimization strategies based on three metaheuristic algorithms; GA, SA and TS, are introduced in Section 3. Section 4 provides an extensive experimental study focusing on performance comparisons between the different heuristic algorithms. Section 5 is the conclusion.

2 WCDMA Radio Network Planning Problems

The problem of WCDMA network planning is basically a p-median problem [15] when only BS location is considered as the decision variable. The p-median problems constitute seeking p number of locations each time regardless of how distant the sites are. We implement a particular reductionism that result in a significant shrinking of the search space. Once the system has been dimensioned, the whole area under consideration can be divided into $p(=K)$ regions, and each region i ($i=1…K$) contains n_i

candidate sites where BSs can be installed. Only one candidate site to install a BS is selected from each region, and an installation cost c_i is associated with each candidate site *i*. With this simplified network scenario, the optimization procedure turns to be more affordable and we can focus on more important issues such as efficient optimization algorithms. The service area is represented by a set of mobile stations (MSs) $M=\{1,\ldots, q\}$.

The problem now is to select one candidate site from each region to install a BS such that the traffic capacity and the number of covered MSs are maximized with the lowest installation cost. We assume that a CPICH signal can be detected if and only if the E_c/I_0 ration is not less than a given threshold γ_0. If one or more CPICH signals are detected, the best server is chosen to be the BS whose CPICH is received by the MS with the highest level. For simplicity, only CPICH and traffic channels are considered. Likewise, consider a 2-way SHO, i.e., one MS connects to two BSs (SHO instances with more than 2 connections can be analysed in a similar way). The following constraints are also taken into account:

1. To be served by a WCDMA network, a MS should receive at least one CPICH signal with an E_c/I_0 that exceeds the threshold value of CPICH signal detection.
2. A MS served by the network must have one and only one best server, whose CPICH signal is received with the highest E_c/I_0 at the MS (without consideration of call admission control—CAC).
3. In downlink, the relative CPICH power is used to determine the SHO server. Therefore, for a MS that is in the SHO state, at least one CPICH signal from a BS rather than its best server should be received by the MS with a power that differs from its best server by no more than a threshold value. This BS will be added into the active set of the MS and selected as one SHO server.
4. Also in downlink, the signals received from the best server and SHO servers are combined at the MS, and the E_b/N_0 requirements of the downlink traffic should be met with consideration to all possible SHO gains in downlink.
5. In uplink, the signals received at the best server and SHO servers are combined at RNC, and the E_b/N_0 requirements of the uplink traffic should be met with consideration to all possible SHO gains in uplink.
6. A BS should connect to a MS as its best server or SHO server; therefore, all SHO links should be taken into account when calculating BS transmission power and the number of MSs served by the BS.
7. APR and PC headroom should be taken into account in the model with consideration to possible SHO gains.

WCDMA radio network planning, which is a multi-objective optimization problem, can be solved as a single-objective problem by combining all aspects of the cost via a weighting function, or can be solved by using the method of Pareto front [11]. We combined the normalized installation cost, the percentage of uncovered MSs, and the percentage of unsupported traffic with a set of weighting factors. Other performance indicators, such as uplink and downlink loading factors, pilot power, signal quality and SHO area etc., can be taken into account with proper weighting factors. To find a set of

candidate sites, with which the cost function achieves the minimum value, is the task of optimization algorithms.

3 Heuristic Optimization Algorithms

In this paper, we have examined and used a simple hill climbing local search algorithm (Greedy) and three well-known meta-heuristics: GA, ESA, and TS algorithms. The SA has been implemented based on a recently proposed approach, which is called evolutionary SA (ESA) [23], while GA and TS algorithms are kept rather standard. The solutions are altered within their neighbourhood via the neighbourhood structures, which are very effective on the performance of exploration heuristics. We identified a neighbourhood structure working based on a totally random selection, which is proved to be robust for different problem instances and can avoid the limitations of some directed exploration heuristics. Therefore, the neighbourhood structure offers moving to a neighboring state by randomly switching to another candidate site.

The Greedy algorithm implemented in this work is used to benchmark the performance of other heuristic algorithms. It does a simple hill climbing local search with the neighbourhood structure in such a way that if the state altered is better than the original one, the move is adopted; otherwise, it is rejected and then another move is tried. Using such a simple algorithm, we test the minimum level of achievable optimization, which will benchmark other heuristic algorithms, and the hardness of the problem instances. If a heuristics algorithm cannot outperform the greedy algorithm, then it has a serious correctness problem.

GA is perhaps the most widely used meta-heuristics in the last two decades. With GA, a population of solutions is adopted and kept evolving so as to achieve a virtually alive environment [14], and genetic operators such as mutation, crossover, selection, and replacement rules are utilized to evolve a population of solutions. A standard GA with lower crossover and higher mutation rates has been implemented in this study. The idea of such genetic operators has been proposed and well discussed in [16] and [17].

SA is another very powerful metaheuristic used in optimization of many combinatorial problems. It is a probabilistic decision making process in which a control parameter called temperature is employed to evaluate the probability of accepting an uphill/downhill move. [13, 18]. Since the probability of promoting a worse state exponentially decays towards zero, it is getting harder to exploit the perturbation facility as the temperature decreases iteration-by-iteration. ESA is a recently developed SA algorithm enhanced with evolutionary operators [19, 20]. The idea is to increase random moves with higher acceptability level of worse states so as to initiate more perturbation options, which are essential for diversifying the solutions. Obviously, a standard SA offers higher acceptance probability in earlier stage of annealing process, while provide very low acceptance probability in the later stages. That makes the process weaker against dynamic and complex problems. To overcome this weakness, ESA offers an iterated procedure in which a rather shorter SA re-operates on the solutions undertaken. Besides, a population of individual solutions can be considered as the undertaken solutions.

TS is also one of the most commonly used meta-heuristics for solving combinatorial problems. The main idea behind TS is to arrange moves to the best of the neighbourhood regardless of the solution quality of the state undertaken. A short and a long term memories are adopted to efficiently enforce the search towards the optimum or a near optimum targeted. The method iteratively keeps searching by using both sorts of memories. Apart from the neighbourhood structure, both sort of memories need well worked out regarding the size of each one. This is to control the efficiency of the algorithm implemented. For more details see [21 – 23]. In this paper, we implemented a standard TS algorithm, which carries out the moves with the random neighbourhood structure introduced above.

4 Experimental Study

Based on our programming model well-discussed in [24], we developed a static simulator of WCDMA networks to evaluate the performance of different optimisation algorithms. An instance of the area to set up a WCDMA network is shown in Figure 1. We consider a rectangular area of 18km×16km to contain K = 19 base stations with 3-sector antennas, i.e. 57 cells in total. All 3-sector antennas are installed with the azimuth being 0^o offset from the North and 0^o down tilt. For each BS, $n_i = n$ =5 candidate sites are made available, i.e. 95 candidate sites in total, from which only one site will be selected to install a BS. 3600 mobile stations (MSs) are uniformly distributed in this area, and all MSs have a traffic activity factor of 1.0.

The set of candidate sites, with which the cost function gets the minimum value, is searched by the heuristic algorithms. Regarding the network shown in Figure 1, the search space is huge as the number of all possible options theoretically becomes 5^{19}. However, the candidate sites are located suitable to be clustered; therefore, it turns to be easier to implement our approach to reduce the complexity of the problems. In order to model a macro-cellular environment, the propagation gain are calculated by using Cost-231-Hata model for urban macro-cell [1], which is given by

$$g = 69.55 + 26.16Log(f) - 13.82Log(h_B) - [(1.1Log(f) - 0.7)h_m - (1.56Log(f) - 0.8)] + [44.9 - 6.55Log(h_B)]\log(d) \quad (1)$$

where f is the signal frequency in megahertz, h_B and h_m are the heights of the antenna of BS and MS in meters, respectively, and d is the distance between BS and MS in kilometres. In this study, f=1950MHz, h_B=25m, and h_m=1.5m. The remaining parameters are set as table 1.

Similar to [2], the installation costs for all candidate sites are kept the same, and assumed all traffics are 12.2kbps voice. The cost function is simplified to the percentage of uncovered MSs, that is:

$$\min\left\{1 - \frac{\text{Total number of uncovered MSs}}{\text{Total number of MSs}}\right\} \quad (2)$$

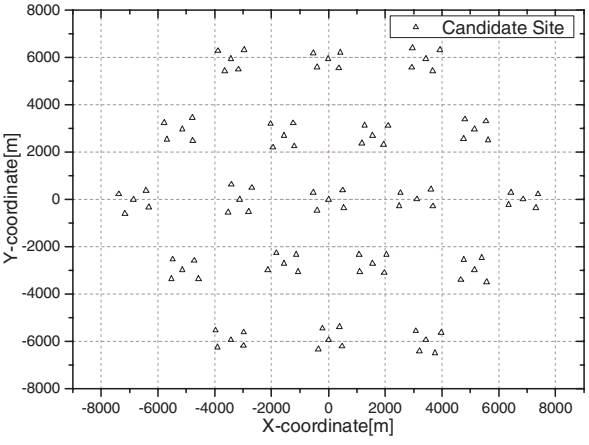


Fig. 1. The instance area to be considered for a WCDMA network

Each time the heuristic algorithm offers a set of candidate sites to install BSs, and then the Monte Carlo simulation is recalled to calculate the cost function value. We set a threshold, denoted as n_s, for the times that Monte Carlo simulation is recalled. When the threshold is met, the algorithm stops immediately. Since each heuristic algorithm has several parameters, parameter tuning is necessary for each algorithm. The parameters are tuned according to the instance shown in Figure 1.

Table 1. Network parameters used in the simulation

Traffic	12.2kbps voice	Max. BS Tx power per link	40dBm
Max. BS Tx power	43dBm	Max. MS Tx power	24dBm
CPICH power	30dBm	Threshold of CPICH E_c/I_0	-18dB
UL E_b/N_0 requirement	4.0dB	DL E_b/N_0 requirement	9.0dB
DL orthogonality	0.7	Noise power density	-174dBm/Hz

Figure 2 (a) and (b) show all the results obtained from the four algorithms with 6000 iterations (n_s=6000), which is normally a very small number of iterations for heuristic algorithms to offer a reasonable solution. However, due to the time consumption of simulation, we have to obtain reasonable solutions within such a number of search. Each experiment was repeated 100 times with the same conditions. Thus 100 optimization results are obtained with each algorithm. All the results were sorted in ascending order for a better display in Figure 2(a).

The best performance is obtained with TS as Figure 2 (a) shows that TS hits the best solution (0.255) for 25 times out of 100 runs, while ESA hits it 19 times, i.e. 6% less than TS, Greedy does 8 times, and GA gets it with an even smaller number (4 times). Although we performed a light parameter tuning and use a high mutation rate (0.9) and

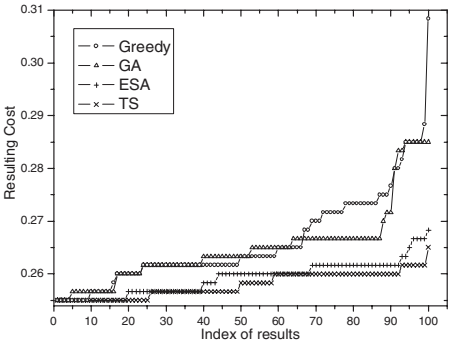


Fig. 2(a). Results of heuristic algorithms; sorted in ascending order (n_s=6000,)

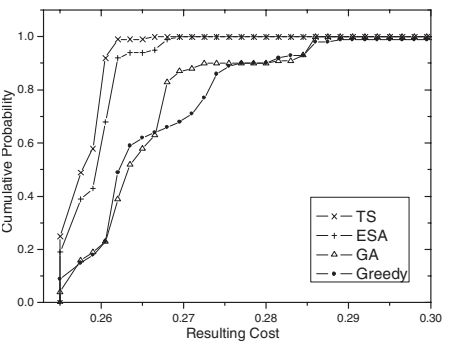


Fig. 2(b). The cumulative probability of the results (n_s=6000)

a low crossover rate (0.1), GA still does not perform well. The reason to keep the mutation rate high and crossover low is to provide more efficient search as crossover reproduces new results based on the parents, which retains the limitations of the parents in children, but mutation breaks such limitations. This always happens when the problems are represented with sets of integers, but may not happen with binary representation, which brings forward some other disadvantages [16,17].

Obviously, 6000 iterations are very low for getting some impressive results from GA. On the other hand, the success of an algorithm cannot be fairly measured only with hitting the best results, but also with the whole set of provided results. In that sense GA does better than Greedy as GA produced 13 results over 27.5, but Greedy did 29. For TS, only 1 result is slightly bigger than 0.265, which shows that TS has a very strong ability to get the optimum or near-optimum solutions even with a small number of iterations. As a well studied SA implementation, ESA provides very impressive but not as good as TS does. ESA has 6 results that are bigger than 0.265 while TS produced only 1 result over that value. This performance reveals that ESA needs further iterations and further investigations to achieve better performance.

For a better understanding of the experimentation, we calculated cumulative probabilities of the results to analyse the distribution of the cost values as shown in Figure 2(b). As expected, TS has the highest probability to get the best solution, as it can get a result less than 0.26 with a probability of 0.9. ESA can get a result which is less than 0.26 with a probability of 0.6, which proves a significant difference with TS. On the other hand, GA provides with 80% of results bigger than 0.26, and almost 40% of them bigger than 0.265. It does much worse than both TS and ESA, because convergence speed of GA is too slow to reach the near optimum solution within 6000 iterations. Greedy algorithm provides slightly worse than GA in general, but it becomes slightly better sometime. There is a clear cross point between the curves of GA and Greedy at the result of 0.266. If the required cost is set to be smaller than 0.266, Greedy will meet this requirement with a slightly higher probability than GA, otherwise, GA will perform better than Greedy.

As 6000 iterations are so small to get reasonable solutions out of search algorithms, we carried out another series of experimentation with more iterations (n_s=10000). That is considered as the medium level of iteration number for heuristic algorithms. All experiments were repeated 100 times for each algorithm. TS is still the best algorithm in terms of the number of hitting the best solution, which is 38 out of 100 runs, On the other hand, ESA obtains the best solution 31 times, which is 7% less than TS. In addition, TS results not bigger than 0.260 at all, while ESA produced 13 times. This concludes that both TS and ESA improve the performance with more iterations, and the performance significantly remains in favour of TS over ESA. However, both algorithms are still comparable.

GA yields the best solution 12 times (it was 3 with n_s=6000), which is 3 times more than the search with n_s=6000, while Greedy did 9 times, which is only 1 more than the previous case. Therefore, the performance of GA is improved significantly this time, while only small improvement was indicated with Greedy. The results are presented in the cumulative probabilities as shown in Figure 3(a), where TS holds the highest probability to provide the best solution. In fact, the probability of hitting a result that less than 0.26 becomes 1.0. On the other hand, ESA performs slightly worse than TS as the probability of 1.0 can be achieved barely after 0.26.

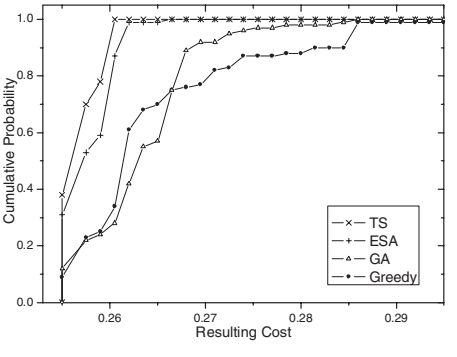


Fig. 3(a). Cumulative probability of the Results (n_s=10000)

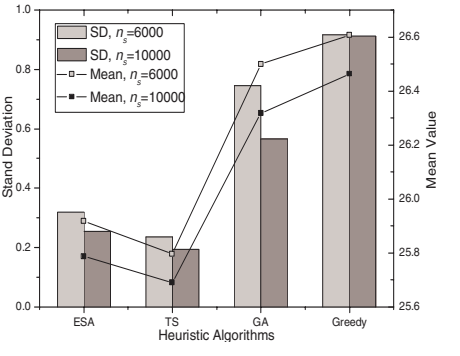


Fig. 3(b). Means and standard deviations of the results with n_s=6000 and n_s=10000. The y-axis right hand is used as the vertical coordinate for mean value.

The results with GA and Greedy remain the same as happened earlier, as GA catches the probability of 0.9 and 1.0 much earlier than Greedy does. However, there are a couple of cases where Greedy performs better than GA. Figure 3(b) indicates the means and the standard deviations (SD) of the results. It shows that TS achieves the lowest means and SD values, while ESA is slightly worse, but other two are much worse. The worst one is expectedly Greedy algorithm in both measures. This means that, TS and ESA optimize as robust as possible, while GA and Greedy do not provide so robust performance, whereas GA does better than Greedy. Another conclusion can be drawn is the contribution of more iterations to the improvement of the performance.

Apparently, Greedy can not benefit of more iterations for robustness, as the SD values almost does not change but the means improved. The other three heuristic algorithms have taken the opportunity of further investigations.

5 Conclusions

WCDMA networks involve much more complicated air interface than the 2G cellular networks, and need to be rigorously optimized to achieve an outstanding performance. In this paper, we considered a comprehensive mathematical model, which consists of several important WCDMA characteristics such as power control, SHO and CPICH power. We optimized the network performance with respect to the BS location by using four heuristic algorithms, namely TS, ESA, GA and greedy search. Two levels of search, 6000 and 10000 iterations, are carried out. The experiments show that TS achieved the best performance and outperformed the other heuristics, while the worst performance appeared with Greedy algorithm, which was expected. ESA did slightly worse than TS, but GA was strongly significantly worse, even with the last search. This is an ongoing project, which investigates for a better modelling of WCDMA network and a better optimisation of the networks. Therefore, we carry out our research in many directions.

References

[1] J. Laiho, A. Wacker, and T. Novosad, “Radio Network Planning and Optimization for UMTS (2nd Edition)”, John Wiley & Sons Inc., Dec. 2005.
[2] E. Amaldi, A. Capone, and F. Malucelli, “Planning UMTS Base Station Location: Optimization Models With Power Control and Algorithms”, IEEE Trans. Wireless Communications, Vol. 2, No. 5, Sep. 2003, pp. 939-952.
[3] R. Mathar and M. Schmeink, “Optimal Base Station Positioning and Channel Assignment for 3G Mobile Networks by Integer Programming”, Annals of Operations Research, 2001, pp. 225-236.
[4] A. Eisenblätter, A. Fugenschuh, et al, “Integer Programming Methods for UMTS Radio Network Planning”, Proc. of WiOpt'04, Cambridge, UK, 2004.
[5] R. G. Akl, M. V. Hegde, et al, “Multicell CDMA Network Design”, IEEE Trans. Veh. Technol., Vol. 50, No. 3, May 2001, pp. 711-722.
[6] C. Y. Lee and H. G. Kang, “Cell Planning with Capacity Expansion in Mobile Communications: A Tabu Search Approach”, IEEE Trans. Veh. Technol., Vol.49, No.5, Sep. 2000, pp. 1678-1691.
[7] S. Hurley, “Planning Effective Cellular Mobile Radio Networks”, IEEE Trans. Veh. Technol., Vol.51, No.2, Mar. 2002, pp. 243-253.
[8] Kocsis, L. Farkas, and L. Nagy, “3G Base Station Positioning Using Simulated Annealing”, Proc. of IEEE PIMRC’02, Vol.1, Sep. 2002, pp. 330-334.
[9] I. Demirkol, C. Ersoy, M. U. Caglayan, and H. Delic, “Location Area Planning in Cellular Networks Using Simulated Annealing”, Proc. of IEEE INFOCOM’01, Anchorage, April 2001.

[10] L. Farkas, I. Laki, and L. Nagy, "Base Station Position Optimization in Microcells Using Genetic Algorithms", Proc. of IEEE ICT'01, Bucharest, Romania, Jun. 2001.
[11] L. Raisanen, and R. M. Whitaker, "Comparison and Evaluation of Multiple Objective Genetic Algorithms for the Antenna Placement Problem", Mobile Networks and Applications, No. 10, 2005, pp. 79-88.
[12] S. B. Jamaa, Z. Altman, et al, "Manual and Automatic Design for UMTS Networks", Mobile Networks and Applications, No. 9, 2004, pp. 619-626.
[13] V. Laarhoven, P. J. M. and E. H. Aarts. "Simulate Annealing: Theory and Applications". Dordrecht, Holland: D. Reidel, 1987.
[14] C. R. Reeves and J. E. Rowe, "Genetic Algorithms-Principles and Perspectives", Kluwer Academic Publishers, 2004.
[15] O. Alp, Z. Drezner, and E. Erkut, "An Efficient Genetic Algorithm for the p-Median Problem," Annals of Operations Research, Vol.122, No. 1-4, September 2003, pp. 21-42.
[16] P.C. Chu and J. E. Beasley, "A Genetic Algorithm for the Set Covering Problem", European Journal of Operational Research, Vol.94, 1996, pp. 392-404.
[17] K. Y. Chan, M. E. Aydin, and T. C. Fogarty, "Parameterisation of Mutation in Evolutionary Algorithms Using the Estimated Main Effect of Genes", Proc. of the IEEE International Congress on Evolutionary Computation, 19-23 Jun. 2004, Portland, Oregon, USA, pp. 1972-1979.
[18] S. Kirkpatrick, C. D. Gelatt Jr., and M. P. Vecchi, "Optimization by Simulated Annealing", Science, Vol. 220, No. 4598, 1983, pp. 671-680.
[19] M. E. Aydin and T. C. Fogarty, "A Distributed Evolutionary Simulated Annealing for Combinatorial Optimisation Problems", Journal of Heuristics, Vol. 10, No. 3, May 2004, pp. 269-292.
[20] V. Yigit, M. E Aydin, and O. Turkbey, "Solving Large Scale Uncapacitated Facility Location Problems with Evolutionary Simulated Annealing", International Journal of Production Research, 44(22), pp. 4773-4791.
[21] F. Glover and M. Laguna, "Tabu search", Hingham, MA: Kluwer Academic Publishers; 1997.
[22] F. Glover, "Parametric Tabu-search for Mixed Integer Programs", Computers & Operations Research, Vol. 33, No.9, Sep. 2006, pp. 2449-2494.
[23] M. Sun, "Solving the Uncapacitated Facility Location Problem Using Tabu Search", Computers & Operations Research, Vol. 33, No.9, Sep. 2006, pp. 2563-2589
[24] J. Yang, M. E. Aydin, J. Zhang and C. Maple, "UMTS Radio Network Planning: a Mathematical Model and Heuristic Optimisation Algorithms", accepted for publication in IET Communications, 2007.

Design of a User Space Software Suite for Probabilistic Routing in Ad-Hoc Networks

Frederick Ducatelle[1,2], Martin Roth[1], and Luca Maria Gambardella[2]

[1] Deutsche Telekom Laboratories (DTL)
Martin.Roth02@telekom.de
[2] Istituto Dalle Molle di Studi sull'Intelligenza Artificiale (IDSIA)
frederick@idsia.ch, luca@idsia.ch

Abstract. We describe the design of MagAntA, a software suite for the implementation of probabilistic routing in ad hoc networks under Linux. MagAntA is written in C and runs completely in user space. This, together with its modular structure, makes it easy to adapt and extend with new algorithms. MagAntA makes use of the Ana4 framework [3], a set of kernel modules that provide the necessary functionalities to support ad hoc mesh networking and facilitate integration with the Linux routing protocol stack. A new version of Ana4 presented in [25] passes each data packet up to user space for routing purposes. Building on this architecture gives MagAntA the possibility to have complete control over routing in user space, so that the per-packet stochastic forwarding typical for probabilistic routing can easily be implemented. MagAntA can also be used in other types of networks such as traditional wired networks, and can easily be extended to incorporate different types of routing algorithms, other than probabilistic ones.

1 Introduction

Wireless mesh networks (WMNs) [1] are telecommunication networks that consist of two types of nodes: mesh clients and mesh routers. Mesh clients are simple devices, usually mobile and often with only one wireless interface. They can serve both as end points of data traffic and as routers to forward data of other nodes. Mesh routers are more powerful devices, equipped with many different wireless interfaces, and are usually not mobile. Mesh routers form a wireless backbone for the WMN. An example of a WMN is the Magnets testbed [7]. An important characteristic of WMNs is their ad-hoc nature. While the placement of mesh routers usually involves some planning, mesh clients can be added, moved, and removed at any time; the resulting network topology can be highly dynamic. WMNs are closely related to mobile wireless ad-hoc networks (MANETs) [11].

The characteristics of WMNs give rise to key challenges that are not present in traditional telecommunication networks. WMN networking algorithms should be adaptive, robust, efficient, and decentralized. The last decade has seen considerable interest in WMNs, especially the routing aspects. A large number of new routing protocols have been developed, such as Ad-Hoc On-Demand Distance

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 121–128, 2007.
© Springer-Verlag Berlin Heidelberg 2007

Vector routing (AODV) [8] and Optimized Link State Routing (OLSR) [4], both of which have been standardized. The field of WMN routing has also been of interest for artificial intelligence researchers. The challenges of an adaptive, robust, and decentralized algorithm fit nicely with the typical properties of distributed processes in nature. The foraging behavior of ants has been a prime source of inspiration. A number of nature-inspired algorithms have been proposed, offering good performance compared to more traditional techniques. Examples include AntHocNet [6], Termite [10] and BeeAdHoc [13]. These algorithms contain features such as continuous exploration of the network, end-to-end path sampling, and stochastic data forwarding. Because of their ample use of probabilistic elements, we refer to them as probabilistic routing algorithms.

Research in WMNs is usually carried out in simulation studies, but the abstractions and simplifications can have a significant impact on the results. Observed results often diverge substantially from what can be expected in reality [5,12]. Researchers are increasingly interested in the use of real world testbeds to support research in WMNs [2,7]. The development of such a testbed is not an easy task, requiring significant technical effort.

The work described in this paper is aimed at facilitating this technical effort. We present MagAntA[1], a software suite for the deployment of probabilistic routing protocols in WMNs using Linux. MagAntA consists of a daemon running in user space. For integration with the Linux kernel, it relies on the Ana4 architecture originally presented in [3]. This is a set of kernel modules that provide the functionality needed to set up a WMN, and ensure a smooth integration with the networking protocol stack. The version of Ana4 proposed by Schioeberg in [25] gives the possibility to send each data packet up to user space with limited extra overhead. Using this system allows the MagAntA system to take complete control over the routing of each data packet, so that the per-packet stochastic forwarding decisions used in probabilistic algorithms can easily be implemented. The location of the routing code in user space and its modular structure makes it easy for additional algorithms to be inserted or modified.

The rest of this paper is organized as follows. First, we provide general scientific background information needed for the understanding of our work. Subsequently, we give a general overview of MagAntA and explain how it is situated with respect to the Linux protocol stack and the Ana4 system it relies on. Then we discuss the MagAntA system itself.

2 Scientific Background

2.1 Probabilistic Routing Algorithms

The probabilistic routing protocols proposed to date are essentially probabilistic distance vector protocols. Information about the network is acquired by sampling paths with control, or ant, packets. These lightweight agents are generated

[1] The name MagAntA is derived from the color magenta of the DTL logo and the ants that formed the initial inspiration for probabilistic routing algorithms.

concurrently and independently by each node, with the task of exploring a path to an assigned destination. An ant going from source to destination collects information about the cost of its path (e.g. end-to-end delay), and then returns this information on the reverse path. Routing tables are updated with the relative utility of using a neighbor to arrive at a destination. These utilities influence routing in the forward direction, giving higher probability to those next hops that have higher utility. Given the nature-inspired origins of these algorithms, the routing table entries play the role of artificial pheromone values in the ant learning process. The routing tables are therefore also called *pheromone tables*, and their entries *pheromone values*. The continuous generation of ants results in information about every path in the network, each node with an estimated measure of path quality. These paths are used to route data packets. Like the ants, data packets are routed *stochastically*, choosing those links associated with higher pheromone values with higher probability. Data for the same destination is adaptively spread over *multiple paths* (but with a preference for the best paths), resulting in *load balancing*.

A variety of probabilistic protocols have been proposed. Many of these are ant inspired algorithms, such as the AntNet [14] algorithm for wired networks and AntHocNet [6] and Termite [10] for wireless ad-hoc networks. Other algorithms include BeeAdHoc [13], inspired by bee swarming behavior, and Q-Routing [15], a reinforcement learning approach. The latter is not nature-inspired, but follows similar principles.

2.2 Deploying a WMN

When deploying a WMN, one has to deal with a number of important issues. A first of these is addressing. WMN nodes should be able to identify themselves even before IP addresses are assigned. This is because local connectivity to a server providing IP addresses cannot be guaranteed. A second issue is the integrated use of the different wireless interfaces available on each node. E.g., a node which has a Bluetooth interface and an 802.11 interface will have different neighbors on each of these two interfaces; it takes part in two different networks. These networks should be integrated to form one WMN. A third issue is packet forwarding, which includes broadcasting, multicasting and unicasting. WMN routing algorithms should be integrated smoothly into the existing TCP/IP protocol stack. Other issues include internet connectivity, vertical hand-over and scalability [3]. An important aspect for any system addressing these issues is that it should require minimal or no changes to existing operating systems and firmware, in order to provide easy installation and portability.

A number of approaches to implement WMN routing functionality have been described in the literature. These include rerouting packets around the existing routing stack to a user space program (AODV-UCSB [20]), rerouting packets to a kernel module (BeeAdHoc [22]), implementing mesh routing functionality into the existing Address Resolution Protocol (ARP) [23], inferring routing requirements based on the behavior of standard ARP (MadHoc [21]), or inserting a new communications layer between the medium access control and the routing layers.

The latter approach is known as layer 2.5. Examples include Ana4 [3], Lilith [16], LUNAR [17], and Microsoft's Connectivity Layer [19]. Software routers like Click Router [18] focus their functionality only on the routing layer. This leaves cross layer issues, such as those found in WMNs, difficult or impossible to implement.

3 MagAntA and the Linux Network Protocol Stack

Here, we present an overview of MagAntA and of how it connects with the network protocol stack in the Linux kernel. In the schematic representation of Figure 1, the MagAntA system is indicated in bold. It consists of a routing daemon running in user space. MagAntA relies on the Ana4 ad hoc virtual interface for integration with the Linux kernel. Data packet flow is depicted in the figure with solid arrows, and control packet flow with dashed arrows.

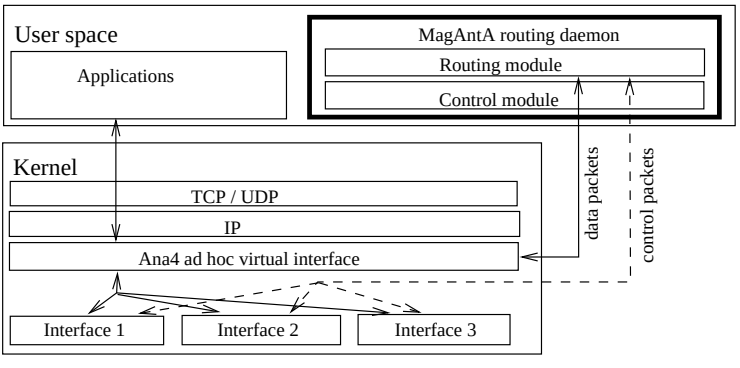


Fig. 1. A schematic overview of the MagAntA architecture

Ana4, proposed in [3], is a layer 2.5 framework for deploying WMNs under Linux. It establishes an ad-hoc layer between the link layer (layer 2) and the IP layer (layer 3). All data packets entering or leaving the WMN pass through this new layer, where they are treated for WMN related issues such as addressing and routing. The integration of the Ana4 ad-hoc layer into the TCP/IP protocol stack is based on the use of a virtual interface. Details about this are outside the scope of this paper. Working with Ana4 provides a number of important advantages for MagAntA. First, it allows an easy integration with the Linux kernel and the existing TCP/IP network protocol stack. Second, Ana4 contains ready solutions for the issues related to WMN deployment that were pointed out in subsection 2.2. Third, since all the code is defined in a number of kernel modules [9], no kernel modifications are necessary, providing easy installation and good portability. Finally, Ana4 can also be used in combination with wired interfaces, allowing us to use MagAntA for the implementation of probabilistic routing in traditional wired networks.

For the integration between Ana4 and MagAntA, we make use of a new version of Ana4, developed by Schioeberg [25]. This version of the architecture allows to send each individual data packet that passes through the Ana4 ad-hoc layer up to user space, where it can be treated by the MagAntA routing software. This possibility to treat each data packet individually inside MagAntA gives a high level of freedom in implementation. This is important when developing probabilistic routing algorithms, which adopt unusual routing practices, such as the use of per packet stochastic routing decisions.

The MagAntA routing daemon consists of a control module and a routing module. The control module provides basic functionality such as communication with the kernel code, scheduling of events, and sending of control packets (which do not pass through Ana4). The routing module contains the routing algorithm. The two modules are connected through a simple interface. The fact that the routing code resides completely in user space and that it has a simple modular structure in which basic functionality is taken care of makes it easy to develop and plug in new routing algorithms. These can be different probabilistic routing algorithms, or more traditional algorithms. This makes MagAntA an easy to use tool for researchers in the field of probabilistic routing. Details about the MagAntA routing daemon are given in the next section.

4 The MagAntA Routing Daemon

In this section, we present the MagAntA user space routing daemon. First, we describe the structure of the program, and then we discuss the probabilistic routing algorithm that is currently implemented in MagAntA.

4.1 The Program Structure

The structure of the routing daemon is shown in Figure 1. It consists of two modules: the control module, which provides basic functionality that is needed for all routing algorithms, and the routing module, which contains the actual routing logic. The modules communicate through a simple routing interface.

The control module provides four basic functions. The first is to gather information about the local host, such as the local ad-hoc address, the MAC address and the available interfaces, and to present this information to the routing module. The second is the transfer of data packets to and from the ad-hoc layer. The control module deals with the technical issues involved in communicating with the kernel, and provides the routing module with pointers to fields in the packet headers that are relevant for routing. The third function is the sending and receiving of control packets. As shown in Figure 1, these do not go through the ad-hoc layer, but are instead sent straight to the relevant interfaces. This is so that specific networks or links can be controlled. The control module opens to this end a raw packet socket. The fourth function is event scheduling. The control module keeps an event list to which the routing module can add and remove events, and calls the routing module whenever an event times out.

The routing interface consists of a structure containing a number of function pointers. Half of these pointers are filled in by the control module, in order to allow the routing module to use the functionality it provides. The other half are filled in by the routing module. They allow the control module to call the routing module when needed, such as at the arrival of a data or control packet, or at an event timeout.

The routing module contains the actual routing code. Since lower level technical issues are taken care of by the control module, this code can concentrate on the algorithm logic. The approach with function pointers in the routing interface makes it easy to plug in different routing algorithms. It is sufficient to provide different functions at the initialization of the pointers to make the control module use different routing code. Currently, we have implemented one probabilistic routing algorithm in MagAntA, which is described in subsection 4.2.

4.2 The MagAntA Probabilistic Routing Algorithm

The probabilistic routing algorithm that is currently implemented in MagAntA is a generic version of the algorithm described in section 2.1. A detailed description is available in [24].

At the start of a communication session, the session's source node establishes an initial path to the destination. To this end, a forward ant is reactively flooded into the network. Upon reaching a node with routing information for the destination, the ant chooses a next hop according to Equation 1.

$$p_{id} = \frac{(P_{id})^{\alpha}}{\sum_j (P_{jd})^{\alpha}} \tag{1}$$

P_{id} is the routing utility, or pheromone, for neighbor i when going to destination d. By raising it to a power α, exploration can be encouraged ($\alpha < 1$) or limited ($\alpha > 1$). When a copy of the forward ant reaches the destination, a backward ant retraces the forward path and updates the routing tables of intermediate nodes. Once the backward ant has returned to the source, data transmission can start.

During the course of the session, the source node proactively improves existing routing information. Proactive forward ants are periodically unicast to the destination, each constructing a path hop-by-hop, taking a new stochastic routing decision at each intermediate node. The probability for a proactive ant with destination d to take a next hop i is defined in Equation 2.

$$p_{id} = (1 - q) \cdot \frac{(P_{id})^{\beta}}{\sum_j (P_{jd})^{\beta}} + q \cdot \frac{1}{N} \tag{2}$$

In this formula, $(1-q)$ is the probability that the ant will be forwarded using the pheromone values. The exponent β modulates exploration. q is the probability that uniform forwarding is applied, and is known as noise. Noise can send packets to neighbors with no pheromone, and is meant to increase exploration. Proactive ants are stopped if for a certain period no data has been sent.

Pheromone is based on round-trip-times (RTTs) experienced by the ants. These RTTs are stored by the forward ants as they traverse the network. The backward ants use the RTTs to update pheromone information in intermediate nodes on the return trip, as shown in Equation 3. As pheromone represents a utility and not a cost, δ_{id} is the inverse of the RTT from the current node through neighbor i to destination d. ρ is a discounting factor.

$$P_{id} \leftarrow \rho \cdot P_{id} + (1 - \rho) \cdot \delta_{id} \tag{3}$$

Data packets are forwarded stochastically, as in Equation 1, but with a larger exponent. This results in a stronger preference for the best paths, but also encourages load balancing.

Links are maintained by periodic hello beacons. Failure to receive these messages, or to overhear any traffic from a neighbor, indicates that the connection is broken. The routing algorithm removes all entries concerning this neighbor from its pheromone table. It does not immediately warn other nodes about the changed situation. If a lost neighbor also causes the loss of a route to a destination, and data packets for that destination continue to arrive, a route loss warning is unicast back to the source.

5 Conclusions and Future Work

We have described the design of MagAntA, a user space software suite for the implementation of probabilistic routing in WMNs. MagAntA builds upon Ana4, a mesh network abstraction layer placed between the medium access and routing layers of the Linux TCP/IP networking stack. Ana4 allows traditional IP routing to take place over a multi-hop mesh network, without the need to change existing communications infrastructure. The fact that MagAntA runs completely in user space and that it has a simple modular architecture makes it easy to adapt and extend with new algorithms, so that it is an easy to use research tool.

Future work will see the continuation of development of MagAntA. Code and documentation will be made available soon. The next step is to perform an extensive testing and verification of the code, as well as performance testing of the routing protocols. Additional routing protocols will also be made available, including both probabilistic varieties and reference algorithms such as AODV. We also want to couple it with a simulator of ad hoc networks, so that results from simulation and testbed deployment can be compared using the same code.

References

1. I. F. Akyildiz, X. Wang, and W. Wang. Wireless mesh networks: a survey. *Computer Networks Journal*, 47:445–487, March 2005.
2. J. Bicket, D. Aguayo, S. Biswas, and R. Morris. Architecture and evaluation of an unplanned 802.11b mesh network. In *Proceedings of Mobicom*, August 2005.
3. N. Boulicault, G. Chelius, and E. Fleury. Ana4: a 2.5 framework for deploying real multi-hop ad hoc and mesh networks. *Ad Hoc & Sensor Wireless Networks: an International Journal (AHSWN)*, 2006. To appear.

4. T. Clausen, P. Jacquet, A. Laouiti, P. Muhlethaler, A. Qayyum, and L. Viennot. Optimized link state routing protocol. In *Proceedings of IEEE INMIC*, 2001.
5. Y. Sasson D. Cavin and A. Schiper. On the accuracy of manet simulators. In *Proceedings of the Workshop on Principles of Mobile Computing (POMC)*, 2002.
6. F. Ducatelle, G. Di Caro, and L.M. Gambardella. Using ant agents to combine reactive and proactive strategies for routing in mobile ad hoc networks. *International Journal of Computational Intelligence and Applications (IJCIA)*, 5(2), 2005.
7. R. Karrer, P. Zerfos, and N. Piratla. Magnets - a next generation access network. In *Proceedings of IEEE INFOCOM*, April 2006.
8. C.E. Perkins and E.M. Royer. Ad-hoc on-demand distance vector routing. In *Proc. of the 2nd IEEE Workshop on Mobile Computing Systems and Applications*, 1999.
9. O. Pomerantz. *The Linux Kernel Module Programming Guide*. iUniverse Inc, 2000.
10. M. Roth and S. Wicker. Termite: Ad-hoc networking with stigmergy. In *Proceedings of Globecom*, 2003.
11. E.M. Royer and C.-K. Toh. A review of current routing protocols for ad hoc mobile wireless networks. *IEEE Personal Communications*, 1999.
12. C. Tschudin, P. Gunningberg, H. Lundgren, and E. Nordström. Lessons from experimental MANET research. *Elsevier Ad Hoc Networks Journal*, 3(2), 2005.
13. H. F. Wedde, M. Farooq, T. Pannenbaecker, B. Vogel, C. Mueller, J. Meth, and R. Jeruschkat. Beeadhoc: an energy efficient routing algorithm for mobile ad hoc networks inspired by bee behavior. In *Proceedings of the conference on Genetic and evolutionary computation (GECCO)*, 2005.
14. G. Di Caro and M. Dorigo. AntNet: Distributed Stigmergetic Control for Communications Networks. In *Journal of Artificial Intelligence Research (JAIR)*, 1998.
15. J.A. Boyan and M.L. Littman. Packet routing in dynamically changing networks: A reinforcement learning approach. In *Advances in Neural Information Processing Systems 6 (NIPS6)*, 1994.
16. V. Untz, M. Heusse, F. Rousseau, A. Duda. Lilith: an Interconnection Architecture Based on Label Switching for Spontaneous Edge Networks. In *Proceedings of ACM SIGCOMM*, 2004.
17. C. Tschudin, R. Gold, O. Rensfelt, O. Wibling. LUNAR - A Lightweight Underlay Network Ad-hoc Routing Protocol and Implementation. In *The 4th Int. Conf. on Next Generation Teletraffic and Wired/Wireless Advanced Networking*, 2004.
18. E. Kohler, R. Morris, B. Chen, J. Jannotti, M. F. Kaashoek. The Click Modular Router. In *ACM Transactions on Computer Systems 18(3)*, 2000.
19. http://research.microsoft.com/mesh/
20. I. D. Chakres, E. M. Belding-Royer. AODV Implementation Design and Performance Evaluation. In *Int. J. of Wireless and Mobile Computing, Issue 2/3*, 2005.
21. F. Lilieblad, O. Mattsson, P. Nylund, D. Ouchterlony, A. Roxenhag. Mad-Hoc AODV Implementation and Documentation. http://mad-hoc.flyinglinux.net
22. H. F. Wedde and M. Farooq and T. Pannenbaecker and B. Vogel and C. Mueller and J. Meth and R. Jeruschkat. BeeAdHoc: an energy efficient routing algorithm for mobile ad hoc networks inspired by bee behavior. In *Proceedings of the conference on Genetic and evolutionary computation (GECCO)*, 2005.
23. S. Desilva and S. R. Das. Experimental Evaluation of a Wireless Ad Hoc Network. In *Proceedings of the 9th Int. Conf. on Computer Communications and Networks (IC3N)*, 2000.
24. F. Ducatelle, M. Roth. Documentation for the MagAntA routing package. Deutsche Telekom Laboratories Technical Report, 2006
25. H. Schioeberg. Routing in ad hoc networks. *MSc thesis, Technische Universität München*, 2006. To appear.

Empirical Validation of a Gossiping Communication Mechanism for Parallel EAs

J.L.J. Laredo[1], P.A. Castillo[1], B. Paechter[2], A.M. Mora[1], E. Alfaro-Cid[3], A.I. Esparcia-Alcázar[3], and J.J. Merelo[1]

[1] Department of Architecture and Computer Technology
University of Granada. ETSIT.
Periodista Daniel Saucedo Aranda s/n.
18071 Granada (Spain)
new@geneura.ugr.es
[2] Centre for Emergent Computing
School of Computing
Napier University
b.paechter@napier.ac.uk
[3] Instituto Tecnológico de Informática
Universidad Politécnica de Valencia (Spain)
evalfaro@iti.upv.es

Abstract. The development of Peer-to-Peer (P2P) systems is still a challenge due to the huge number of factors involved. Validation of these systems must be defined in terms of describing the adequacy of the P2P model to the actual environment. This paper focuses on the validation of the Distributed Resource Machine (DRM) as a computational P2P system when applied to Evolutionary Algorithms (EAs) using exclusively gossip-based mechanisms for communication. The adequacy will be measured by the range in which performance speedup actually takes place. Validation has been carried out by running an empirical performance study based on benchmarking techniques. It shows that it scales only up to a limited and small number of nodes, which is problem-dependent. Furthermore, due to the reason found for this lack of scalability, it seems unlikely that massive scalability takes place.

1 Introduction

Peer-to-Peer (P2P) systems became popular in 1999 with the first release of the file-sharing application Napster[1]. Since then, they have become a serious alternative for applications such as file-sharing, distributed computation and instant messaging, which were difficult or expensive to solve by means of classic methods (i.e. Client-Server) [13]. Problems addressed using P2P systems differ but have certain points in common such as the fact that they are decomposable into subproblems, scalable in size and they need a huge amount of resources to be solved.

[1] http://www.napster.com. Accessed on January 2007.

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 129–136, 2007.
© Springer-Verlag Berlin Heidelberg 2007

From the computer engineering perspective, P2P systems are a challenge to develop due to the diversity of factors involved. First, it is hard for the human mind to think in a distributed and decentralised way, which makes developing a theoretical model non-trivial; and second, it is difficult to shape that model in a computational system like the internet where there is no quality of service guarantees due to questions such as fault tolerance, aggregation of nodes, communication overhead or bandwidth. There is also the problem of having a virtual topology (the so-called overlay network) with its own routing mechanism on top of a physical network; the overlay network should keep a balance between adequacy for a given problem and adaptation to the physical substrate where it is mapped [6].

This large number of factors means that it is not possible analytically to find an optimal solution (in terms of leveraging existing resources, for instance) for these systems but only to reach a practical compromise. In addition to simulations which prove the viability of a model, we need a tool in order to validate the implementation which describes the actual operational compromise between a P2P model and external factors. A tool like this will have to be designed according to the inherent features of the system.

In this paper we try to validate the DRM [2] as applied to parallel EAs via an empirical performance study. The DRM is a computational P2P system that was developed as the communication layer within the Distributed Resource Evolutionary Algorithm Machine (DREAM) project architecture. The DREAM framework focuses on distributed processing of evolutionary algorithms (which accounts for the "EA" in DREAM) using the internet in a P2P scalable fashion by means of the DRM and providing the library JEO [1] for developing parallel EAs. the DRM maintains an unstructured overlay network that uses so-called gossip mechanisms [9,8]. It provides two ways of communication at application level. The first one is "direct communication", which implies that the developer has explicit knowledge and can manage the communications through the use of direct messaging routines (as used in JEO). The second one consists of following a "gossip" mechanism by inheritance of a class called Collective which spreads messages in a probabilistic manner. The use of the Collective has not been yet studied as communication mechanism for parallelising EAs and it is here where this study focuses.

An additional difficulty that the DRM presents is that it was conceived as an experimental implementation of the newscast protocol [10]. Therefore, the DRM does not have a solid theoretical model as its foundation, even though it shares some features of the newscast protocol such as decentralisation, adaptivity and heterogeneity. This makes it quite difficult to extrapolate statistical properties from studied newscast simulations which demonstrate the viability of the model and emulations which validate it [14]. The approach we took to the validation of the DRM Collective for solving parallel EAs consists of an empirical performance study based on benchmarking techniques.

Since there is no possibility of simulating or emulating the DRM, we manage to overcome this limitation in section 2 by using an application benchmark in order

to obtain an empirical study of performance. In this section it is also introduced the way the Collective works and how the gossip communication takes place. Finally, results are presented in section 3 and in section 4 some conclusions are reached.

2 Validation of the DRM Collective for Solving Parallel EAs

The DRM architecture consists of a set of nodes, each one running at least two threads, one supporting the computation (agent) and other supporting the communication (the so-called correspondent). Each correspondent maintains a fixed-sized cache of 100 cache entries. These caches are regularly exchanged in an asynchronous manner after a ΔT time interval.

As explained in section 1, there is neither a well defined model for the DRM, nor for its corresponding algorithm; however, the DRM shares its main design and behaviour features with the newscast protocol. However, the DRM within the DREAM framework [7,2] is a working platform used by the scientific community [3,5] for the development of distributed evolutionary algorithms with the aim of obtaining high performance. The DRM property of scalability contributes to the improvement of performance in a typical highly expensive computation problem by taking advantage of the resources on a multicomputer distributed environment. This context leads us to defining validation in terms of scalability as *the range where a certain increase in speedup is maintained when an experiment is run on an increasing number of nodes* (as opposed to the idea of validation of a well-defined, viable model).

In order to deal with scalability we have developed an application benchmark which consists of several instances of the knapsack problem (extracted from [11]). These instances will form a representative set of experiments with different complexities in the field of evolutionary algorithms. The experiments have been conducted on a Beowulf cluster (described at http://www.inf.ed.ac.uk/systems /beowulf) using sets of up to 32 nodes. Furthermore, all experiments have been considered under an optimistic scenario where the number of nodes remain stable for each execution without leaves or aggregations. Thus, this study focuses on the scalability aspects avoiding fault tolerance and robustness considerations.

Since DREAM is a framework that combines at least two different Computer Science disciplines (Evolutionary Algorithms and Distributed Programming), a representative application benchmark for its communication layer (DRM) will have to deal with both in order to obtain meaningful results.

The application benchmark consists of a classical genetic algorithm that solves the knapsack problem; this benchmark has been chosen since it adapts to the DREAM application target (DREAM was initially intended for distributed evolutionary algorithms), and also because it is easily scalable by changing its size and correlation function. The code is parallelised following the island model [4]. Nine different data sets have been configured by combining different problem sizes and different correlation functions between weight and value in the objects.

The *evolutionary algorithm* [11] is defined by the following configuration decisions: The initial population is generated randomly in the feasible space of solutions. Each individual representing a solution is coded as a binary vector. When a individual is non-feasible a repairing mechanism erases the objects in the vector until a feasible solution is reached. Each individual is evaluated according to the function $\rho(x) = \sum_{i=1}^{n} x_i V_i$, where x is the binary vector, P is a vector containing the values of each object included in the knapsack and n is the number of objects. The sampling mechanism used in each generation is fitness proportional selection with elitism, and the evolutionary operators are 1-point crossover and 1-bit mutation. The algorithm ends when a pre-established solution for each instance of the knapsack problem is reached. Therefore, there is not a fixed number of evaluations by run but the algorithm has to find certain quality in solutions. Since this termination condition is stochastic and dependent on the run, the results are given on the average of 30 independent runs. Additionally, we have a fixed population size of 100, crossover probability at 0.65 and mutation probability at 0.05. These parameters are quite standard and do not have a big influence on the overall performance of the P2P system.

The evolutionary algorithm is parallelised using the *island model* [4]. It takes advantage of the inherent parallelism at population level (coarse grained), as different individuals can be evaluated in different population sets, called *islands*. The communication between islands is defined by the migration of individuals (called migrants) from one island to another. Although the use of this model implies an algorithmic effect on the EA by the variation in the number of islands that we will have to assess in future works. Concerning parameters, the migration rate has been fixed at 5% of the individuals in an island, the migration probability at 0.05. The migration selection and replacement is made through a roulette wheel selector that stochastically selects the best individuals for emigration and the worst individuals for replacement. All communication among the islands is implemented using the interface that the DRM Collective provides, which consists of two methods, getContribution and setContibution; the former has been implemented to get the immigration *to* the island and the latter to set the emigration *from* the island. getContribution would be equivalent to a typical parallel function which receives a message. However, getContribution works asynchronously and in a non-deterministic way. setContribution would correspond to the function send message but once again neither the moment of sending nor the destination can be known.

Finally, the *metrics* we are going to use will be defined by the speedup and the average hops per migrant sent. The *speedup* gives an idea about the scalability in a Computational P2P system and it is described as $S_n = \frac{T_1}{T_n}$, where, S_n is the speedup an experiment reaches when it is solved by n nodes, T_1 is the time spent when solved sequentially in one node and T_n when solved by n nodes.

The *average hops per migrant sent* gives by $Hops_{avg} = \frac{\sum_{i=1}^{N} Hops_{Migrant_i}}{N}$, describes the performance of the routing system in P2P and is the ratio of the sum of the total number of hops incurred by any sent migrant ($Hops_{Migrant_i}$) to the total number of sent migrants (N) [15].

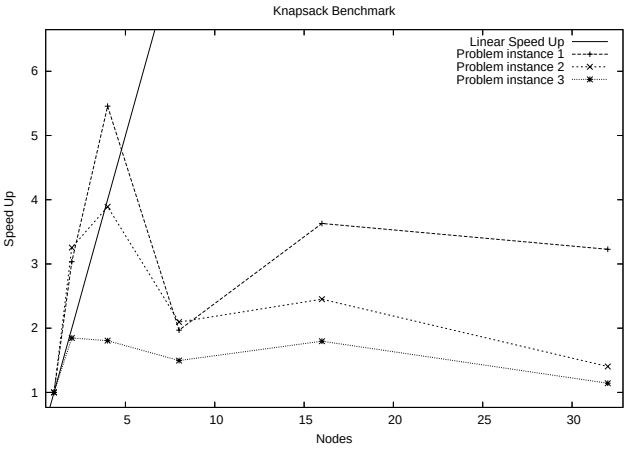


Fig. 1. Scalability of the knapsack problem in terms of speedup when solved by a distributed evolutionary algorithm on the DRM

3 Results

This section presents and analyses the results obtained with the benchmark according to the metrics presented in section 2. It is important to note that the results are the average of 30 independent executions in order to gain accuracy since our benchmark is non deterministic (both evolutionary algorithms and P2P systems behave in a non deterministic way).

Figure 1 depicts three speedup curves corresponding to three instances of the knapsack problem. They represent the best, the worst and one intermediate result within the whole set of problems. The graph shows the common feature of a lack of scalability from 4 nodes onwards. Therefore, this analysis will try to establish the main reasons that cause the decrease in performance in a framework thought to be highly scalable.

The first step towards the analysis consists of determining whether the lack of scalability could be caused by communications (since the major bottleneck in distributed systems is usually the communication overhead). We have monitored, as a relative measure for the consumed bandwidth, the number of migrants sent between islands where each migrant corresponds to a solution exchanged in the migration process through the whole system in the same execution fragment of the algorithm on a configuration consisting of 32, 16 and 8 nodes. As shown in Figure 2, the growth in the number of migrants sent with respect to the system size, points to the mentioned communication overhead as the cause of the lack of scalability.

This approximation will be further proved and defined more precisely by the results showing the number of average hops used by migrants which corresponds to a linear growing with respect to the number of nodes $Hops_{avg} \approx N$, instead of a logarithmic growing of the hops as expected in an Small World system such

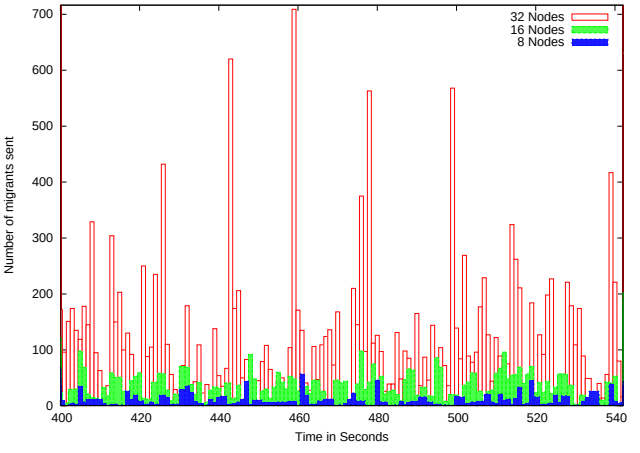


Fig. 2. Number of migrants sent within the whole system in a monitored execution fragment. Note the increase in communications when the system grows.

as newscast [10]. This gives us a hint to the reasons of the lack of scalability: instead of newscasting, the communication mechanism is closer to some kind of asynchronous broadcasting.

However, when we say *some kind* of asynchronous *broadcasting*, actually, we are not actually checking whether each migrant reaches all the nodes (a feature that would define a broadcast system). However, the message Time-to-Live (TTL), expressed as the average hops life, is big enough to make this possible, which describes in terms of scalability the behaviour of a broadcast system.

On the other hand, as mentioned in section 2, the DRM is an asynchronous system. The asynchrony under study shows a correlation between the number of nodes and the number of migrants sent per iteration, where iteration is defined as the execution of a generation within the evolutionary algorithm [11]. On average this correlation is defined by $\frac{MigrantSent}{iteration} \approx 0.1N$, where N is the number of nodes.

Obviously, this correlation will be applicable up to a certain system size. As the system grows, we expect to find an asymptotic bound due to limitations in bandwidth and the inherent communication overhead. Unfortunately, the 32 nodes used in our experiments are not enough to reach it.

As conclusion we can say that the DRM Collective behaves as something close to an asynchronous broadcast system, where broadcasting is depicted by the number of average hops per migrant sent ($Hops_{avg} \approx N$) and asynchronism is defined in average by the correlation function $\frac{MigrantSent}{iteration} \approx 0.1N$. Both facts account for the lack of scalability.

4 Conclusions

According to the definition of validation given in section 2, we can conclude that the DRM Collective is valid up to a range of four nodes when it is used

for solving an evolutionary algorithm; a different problem would yield a different figure. However, the study also proves that, within that limited number of nodes, the DRM can yield good performance, even super-linear speedup, at least when applied to the kind of problems it has been designed for. This implies that the practicality of the DRM Collective for solving the communication problem in parallel EAs can not be extended further than a limited and small number of nodes, at least for problems of this kind, as has already been experienced by some users of this framework, and published in [3,5] (Spanish). In [5] an application of genetic programming to the classification of signals with different spectral densities was used to analyse the speedup capabilities of the DRM. Executions were run with 1, 2, 3 and 4 islands. The results showed that the execution time was reduced by more than half when using 2 islands but the inclusion of more islands did not reduce the execution time as much as expected (sub-linear fashion). These results are consistent with the results presented in this paper.

However, from the perspective of P2P systems, our validation has concluded that the DRM Collective is not really massively scalable for solving parallel EAs. It is difficult to know at this stage what kind of underlying problem can occur or how significant in the EA is the effect of solving the problem with a different set of islands, these questions will have to be addressed in future works. Theoretically, the TTL parameter should govern the number of hops the migrant should travel; by limiting its lifetime, the average number of hops should grow logarithmically and not linearly with network size, as it happens now. However, the TTL parameter seems to have been fixed within the DRM implementation, and changing it has proved impossible for the authors. In theory, too, limiting cache size would also limit the amount of migrants sent across the network; but in practice, we have also shown that the number of migrants remains the same.

Therefore, our future work will focus on the improvement of the DRM by adapting gossip features to the problem of communication in parallel EAs. A possible solution points to exchange a partial number of cache entries or even single one instead of the current whole cache exchange proposed in the newscast protocol. For this, we will try to follow the methodology presented in [14].

Finally, we believe that validation is a necessary step within the development of a P2P system. This paper contributes with a practical case where validation has been carried out with an empirical performance study using benchmarking techniques.

Acknowledgements

This work has been performed under the Project NadeWeb (TIC2003-09481-C04) and partially supported under the Project HPC-EUROPA (RII3-CT-2003-506079) Research Infrastructure Action under the FP6 “Structuring the European Research Area” Programme.

References

1. M. G. Arenas, B. Dolin, Juan-Julián Merelo-Guervós, P. A. Castillo, I. Fernández de Viana, and M. Schoenauer. JEO: Java Evolving Objects. In W. B. Langdon, E. Cantú-Paz, K. Mathias, R. Roy, D. Davis, R. Poli, K. Balakrishnan, V. Honavar, G. Rudolph, J. Wegener, L. Bull, M. A. Potter, A. C. Schultz, J. F. Miller, E. Burke, and N. Jonoska., editors, *Poster Accepted at GECCO 2002*, page 991, 2002. Available from http://geneura.ugr.es/pub/papers/MPP104.ps.gz http://citeseer.nj.nec.com/context/2198210/0.
2. M.G. Arenas, Pierre Collet, A.E. Eiben, Márk Jelasity, J. J. Merelo, Ben Paechter, Mike Preuß, and Marc Schoenauer. A framework for distributed evolutionary algorithms. Number 2439 in Lecture Notes in Computer Science,LNCS. Springer-Verlag, September 2002. Available from http://www.amazon.com/exec/obidos/ASIN/3540441395/perltutobyjjmere and http://www.springeronline.com/sgw/cda/frontpage/0,10735,5-40109-22-224 086-0,00.html.
3. Víctor Martín Molina; Juan Julián Merelo Guervós; Juan Luis Jiménez Laredo; Maribel García Arenas. Algoritmos evolutivos en Java: resolución del TSP usando DREAM. In *Actas XVI Jornadas de Paralelismo, incluido en CEDI'2005. Granada, septiembre 2005*, pages 667–683. Thomson, Septiembre 2005.
4. E. Cantú-Paz. Topologies, migration rates, and multi-population parallel genetic algorithms. In *GECCO-99, Genetic and Evolutionary Computation Conference*, pages 13–17, 1999.
5. Eva Alfaro Cid. Aumento de velocidad de convergencia usando un sistema distribuido (dream). Technical report, ITI, November 2004.
6. Diego Doval and Donal O'Mahony. Overlay networks: A scalable alternative for P2P. *IEEE Internet Computing*, pages 79–82, July 2003.
7. DREAM. Distributed resource evolutionary algorithms machine. http://dr-ea-m.sourceforge.net, 2001–2003. accessed on January 2007.
8. M. Jelasity, M. Preuss, and B. Paechter. A scalable and robust framework for distributed applications. In *CEC 2002*, IEEE Press, pages 1540–1545, 2002.
9. Márk Jelasity, Alberto Montresor, and Ozalp Babaoglu. Gossip-based aggregation in large dynamic networks. *ACM Trans. Comput. Syst.*, 23(3):219–252, 2005.
10. Márk Jelasity and Maarten van Steen. Large-scale newscast computing on the Internet, October 2002.
11. Zbigniew Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs*. ©Springer, 3 edition, November 1998.
12. Ralf Steinmetz and Klaus Wehrle, editors. *Peer-to-Peer Systems and Applications*, volume 3485 of *Lecture Notes in Computer Science*. Springer, 2005.
13. Ralf Steinmetz and Klaus Wehrle. What is this *"peer-to-peer"* about? In *Peer-to-Peer Systems and Applications* [12], pages 9–16.
14. Spyros Voulgaris, Márk Jelasity, and Maarten van Steen. A robust and scalable Peer-to-Peer Gossiping protocol. In Gianluca Moro, Claudio Sartori, and Munindar P. Singh, editors, *AP2PC*, volume 2872 of *Lecture Notes in Computer Science*, pages 47–58. Springer, 2003.
15. H. Zhang, A. Goel, and R. Govindan. Using the small-world model to improve Freenet performance. volume 3 of *Proceedings IEEE INFOCOM 2002*, pages 1228–1237. IEEE Infocom, 2002.

A Transport-Layer Based Simultaneous Access Scheme in Integrated WLAN/UMTS Mobile Networks

Hyung-Taig Lim[1], Seung-Joon Seok[2], and Chul-Hee Kang[1]

[1] Department of Electronics Engineering, Korea University
5-ga, Anam-dong, Sungbuk-gu, Seoul 136-701 Korea
{limht,chkang}@widecomm.korea.ac.kr
[2] Department of Computer Science and Engineering, Kyungnam University
449, wolyong-dong, Masan, 631-701 Korea
sjseok@kyungnam.ac.kr

Abstract. The integration of UMTS cellular network and wireless LANs enables users to achieve both the broad coverage of UMTS cellular network and the higher data rate of WLANs. In this paper, we present a transport-layer based simultaneous access scheme which enhances throughput by efficient use of these networks. Two representative transport protocols in the Internet, UDP and TCP, show significantly different behaviors against end-to-end delay and packet losses, and UMTS cellular network and WLANs have different packet losses and end-to-end delay characteristics. After prediction of UDP and TCP throughput in each network, each transport protocol is allocated to the network where total throughput can be maximized. We evaluate the proposed mechanism using NS-2 and show the improvement of the total throughput.

1 Introduction

With the rapid development of wireless and mobile communications, various radio technologies have appeared. The Universal Mobile Telecommunications System (UMTS) cellular networks have been successfully deployed and provide ubiquitous coverage and high mobility. However, they don't offer high data rates. On the other hand, WLANs provide higher data rates but limited coverage and mobility. Through the integration of these two networks, users can achieve the benefits of both networks.

To provide seamless mobility across these networks, the architecture for the integration of these two networks has been investigated. Proposed architectures can be classified into tightly coupled and loosely coupled scheme according to their integration depth. This integration complexity affects inversely the execution performance of the mobility procedure. Also, these two networks have different QoS mechanisms. To maintain end-to-end QoS in the vertical handoffs, the QoS provisioning across these networks such as QoS mapping and resource allocation have been studied [1]. Another research challenge is to choose the

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 137–144, 2007.
© Springer-Verlag Berlin Heidelberg 2007

proper network and proper times, which is called vertical handoff decision. The early vertical handoff decision algorithm is based on the received signal strength (RSS). The mobile station starts to perform vertical handoff when the RSS of the WLAN is greater than a certain threshold [2]. However, when the WLAN is overloaded, the mobile station cannot obtain throughput enhancements and make unnecessary vertical handoffs. In [3], the mobile station makes a handoff decision after measuring available bandwidths in the candidate WLAN. If the available bandwidth is lower than a certain threshold, the mobile station does not enter the WLAN. In this method, the mobile stations access only a network. [4] proposes a session individual handoff algorithm by which the mobile station can access multiple networks and allocate each session to the most proper network to achieve maximum throughput. The main decision metric of these previous works is the network bandwidth. However, in this paper, we observe that the throughputs of applications are strongly dependent on the kind of underlying transport protocol which shows significantly different behaviors against packet loss and delays. For instance, TCP performance has inverse relationship with end-to-end delay, which does not affect severely UDP performance. Moreover, the packet losses degrade severely the TCP performance than UDP performance. More TCP throughput can be achieved by allocating TCP traffics to the network with the less packet losses and shorter end-to-end delay. Moreover, the UMTS cellular networks and WLANs have different characteristics in end-to-end delay and packet losses as well as bandwidth.

In this paper, we focus on obtaining the maximum throughput with simultaneous access by considering transport-layer throughput as a decision metric. First, we describe the predictions of transport-layer throughputs. With these predictions, we determine traffic allocations which maximize the sum of the transport-layer throughputs. By ns-2 simulations, we evaluate the proposed algorithm and show significant throughput improvement compared with previous works.

The rest of the paper is structured as follows. Section 2 introduces the proposed access scheme. In section 3, we discuss the performance of each transport protocol. We evaluate the proposed algorithm through the simulations in section 4. We conclude the paper in Section 5.

2 Transport-Layer Based Simultaneous Access Scheme

In this section, we propose transport-layer based simultaneous access scheme, present its mathematical formulation, and describe procedures for the proposed algorithm.

The bandwidth of the network has mainly been considered as a decision metric in previous proposals for vertical handoffs. However, we consider transport-layer throughput as a decision parameter, since the throughputs of applications are strongly dependent upon the kind of their underlying transport protocol. Two transport protocols in the Internet, UDP and TCP, were designed for different purposes. TCP has the congestion control and the error recovery mechanism,

but UDP doesnt have. Therefore, the packet loss and end-to-end delay cause different effects to the two transport protocols. As the packet loss rate increases, TCP is more severely damaged by the packet loss than UDP, since TCP considers a packet loss as the indication of network congestion and multiply decreases its congestion window. Also, the TCP throughput has the inverse relation ship with end-to-end delay. So, in a given bandwidth, two transport layer protocols may have significant different throughputs. Moreover, WLANs and the UMTS cellular networks are different in end-to-end delay and packet losses as well as bandwidths. Usually, WLANs have lower delays, since their architectures are simpler than that of the UMTS cellular networks. We can obtain higher throughput by allocating a traffic to the network where the traffic can achieve better throughput. With multiple flows, we assign each flow to the appropriate network in order that the sum of transport layer throughputs can be maximized. This scheme can be expressed as follows:

Given:

N : the number of flows on the mobile station

M : the number of networks which the mobile can access

THP_i^j: the transport layer throughput of the flow i in the network j

B^j : the available bandwidth in the network j

Find:

$$n_{ij} = \begin{cases} 0, \text{ if flow i is not assigned to the network j} \\ 1, \text{ if flow i is assigned to the network j} \end{cases}$$

Maxmimize:

$$\sum_{i=1}^{N} \sum_{j=1}^{M} n_{ij} THP_i^j \tag{1}$$

Subject to:

$$\sum_{i=1}^{N} THP_i^j \leq B^j, \forall j$$

$$\sum_{i=1}^{N} n_{ij} = 1, \forall j$$

The operation of proposed process is summarized in Fig. 1. For the proposed method, we define two states of a mobile station: a single access state and a simultaneous access state. The single access state is set when the mobile station can only access UMTS cellular networks. Otherwise, the simultaneous access mode indicates that the mobile station can access WLANs and UMTS cellular networks. Initially, the mobile station accesses the UMTS cellular networks. As mentioned previously, the mobile station monitors WLAN signals to find WLANs. When a WLAN signal is detected, the mobile station measures the RSS of the WLAN (RSS_{WLAN}) to determine whether WLAN links can be used. If RSS_{WLAN} is lower than the predetermined threshold value (THR_{WLAN}), a mobile station in the simultaneous access changes all ongoing sessions to the UMTS cellular networks and sets its state as single access. On the other hand,

if RSS_{WLAN} is greater than THR_{WLAN}, the mobile station probes available bandwidth of the WLAN (BW_{WLAN}) to extract achievable throughput. The mobile station in simultaneous access state does not need probe available bandwidth, since it already knows the available bandwidth. The available bandwidth is dependent on the load of WLAN and the wireless link status. Also, in this step, we assume the mobile station knows available bandwidth of UMTS cellular networks since it is assumed that it can always access the UMTS cellular networks. With two available bandwidths, we predict achievable throughputs in each network and find traffic allocations which can maximize the sum of transport layer throughputs as mentioned previously. Finally, we allocate each traffic by maximization configurations.

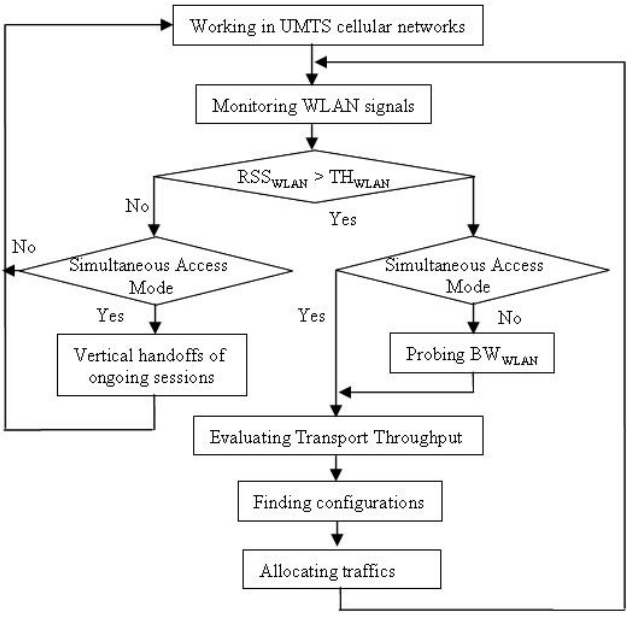


Fig. 1. The flow diagram of transport-layer based simultaneous access scheme

3 Evaluation of Transport Layer Throughputs

In order to find traffic allocations, the throughput of each transport protocol needs to be predicted. In this section, we describe the performance of each transport protocol.

As described previously, the UDP does not have error control and flow control mechanisms. Hence the sending rate of UDP flows is not affected by packet errors and UDP throughput can be expressed as follows:

$$THP_{UDP}(p) = THP_{UDP}(0) \times (1 - p) \tag{2}$$

where p and $\mathrm{THP}_{UDP}(\mathrm{p})$ are the packet loss rate and the UDP throughput with the packet loss rate p.

However, the TCP throughput is severely dependent on RTT and the packet loss rate. TCP undergoes two phases: slow start phase and congestion avoidance phase. The TCP throughputs during these phases have been dealt in many articles. Wireless networks usually have limited bandwidths, so the PFTK model [5] may not be proper in the wireless environments. Hence, we adopt the methods and models which are presented [6, 7]. In these works, the system is modeled as a single bottleneck capacity μ packets/second and an overall delay τ. In this model, the congestion window in the congestion avoidance mode grows linearly until it reaches $\mu(\tau+1/\mu)$ and afterwards it grows sub-linearly.

To evaluate the throughput in slow start mode, duration of slow start phase T_{ss} and the number of packets during the slow start N_{ss} are needed. Their values are expressed as follows [7]:

$$T_{ss} = (log_2^{W_{ss-thresh}} + 1) \times T \tag{3}$$

$$N_{ss} = (W_{ss-thresh}^2 - 1) \tag{4}$$

where $W_{ss-thresh}$ is the slow-start threshold and T denotes the overall delay τ plus the service time $1/\mu$.

On the other hand, the throughput in the congestion avoidance mode is expressed as follows [6] :

$$\overline{\lambda} = \left[\int_o^{t_A+t_B} p(1-p)^{n(t)} \frac{n(t)}{t} \lambda(t) dt\right] + \left[(1-p)^{N_{max}} \frac{N_{max}}{t_A + t_B}\right] \tag{5}$$

where p is the random packet loss rate, n(t) is the number of packets by time t, $\lambda(t)$ is the instantaneous instant throughput, N_{max} is maximum number of packets until a packet loss, t_A is linear growth time and t_B is sub-linear growth time.

However, this may not be proper to be used for the proposed method, since it is too complex for a mobile station to evaluate TCP throughput fast and causes the mobile station to execute the numerical integration. Hence, we use periodic evolution of congestion window for simplicity. The number of packets transmitted until the next packet loss can be thought as 1/p. Let the congestion window size at the loss event W, the relationship between the packet loss rate p and the congestion window size W can be expressed as follows :

$$\frac{1}{p} = \frac{1}{2}\left(W + \frac{W}{2}\right)\left(W - \frac{W}{2}\right) \tag{6}$$

Using Eq. (6), we can obtain the window size (W/2) at the start of a periodic cycle and the congestion window size at the loss event. With these value, the time to next packet loss can be found and it contains a linear increase time t_A and a sub-linear time t_B. Congestion window growth rate is expressed as follows [6]:

$$\frac{dW}{dt} = \begin{cases} \frac{1}{T}, & W < \mu T \\ \frac{\mu}{W}, & W \geq \mu T \end{cases} \tag{7}$$

By integration of Eq. (7), t_A and t_B can expressed as follows:

$$t_A = \begin{cases} \frac{W}{2T}, & 0 < W \leq \mu T \\ \frac{1}{T}(\mu T - \frac{W}{2}), & \mu T < W \leq 2\mu T \\ 0, & W > 2\mu T \end{cases} \tag{8}$$

$$t_B = \begin{cases} 0, & 0 < W \leq \mu T \\ \frac{(W)^2 - (\mu T)^2}{2\mu}, & \mu T < W \leq 2\mu T \\ \frac{1}{2\mu}((W)^2 - (\frac{W}{2})^2), & W > 2\mu T \end{cases} \tag{9}$$

Hence, the TCP throughput considering both slow start and the congestion avoidance can be presented as follows:

$$THP_{TCP} \approx \frac{1}{T_T}\left(N_{SS} + \frac{T_T - T_{SS}}{p(t_A + t_B)}\right) \tag{10}$$

where T_T is total TCP running time.

4 Simulation Results

To evaluate our proposed model, we use the ns-2 [8] and the EURANE [9] package and tightly coupled architecture. We assume that the bandwidth of UMTS is 384 kbps and the end-to-end RTT over UMTS is 300ms, and the bandwidth of WLAN is 1Mbps and end-to-end RTT is 130ms. For TCP traffics, we use long-lived traffic. In case of short TCP traffics such as HTTP, traffic duration is too short to gain the throughput enhancement with a vertical handoff and it may be better not to execute the vertical handoff.

We first simulate the proposed algorithm with a FTP flow. When a mobile station finds the WLAN, it can have various packet loss rates according to its wireless channel condition, hence the simulation is executed for various packet loss rates in WLAN and the fixed average throughput in UMTS. The result for this scenario is shown in Fig. 2. In this scenario, the proposed algorithm changes the connected network from WLAN to UMTS in the region where the predicted throughput is higher than the UMTS throughput, otherwise, it sticks to UMTS networks, since the propose algorithms choose the network where a mobile station can achieve higher throughput. However, the RSS based method and muti-net optimized algorithm changes the attached networks to the WLAN even in the region where the throughput in the WLAN is degraded, since they do consider only RSS or bandwidth.

Simulation results for a constant bit rate (CBR) flow and a FTP flow are shown in Fig. 3. We simulate the proposed algorithm for various CBR rates. In this scenario, the proposed algorithm allocates the CBR traffic to UMTS cellular networks and the FTP traffic to WLANs until the bandwidth of UMTS cellular networks is greater than CBR request, since allocation of both flows to WLAN causes throughput degradation due to the competition between both flows. Hence, when CBR request is less than UMTS link capacity, only the FTP

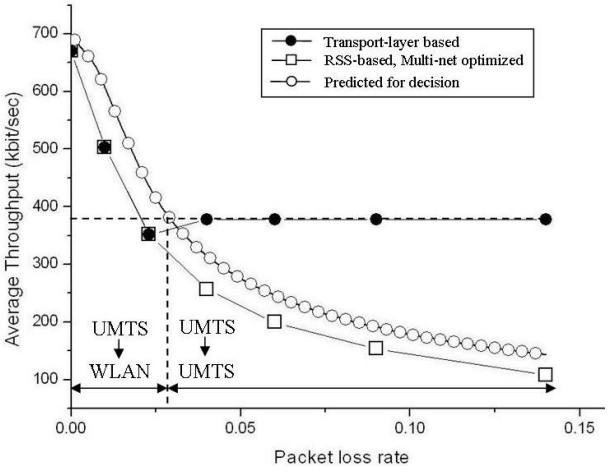


Fig. 2. The TCP throughput with only FTP traffic

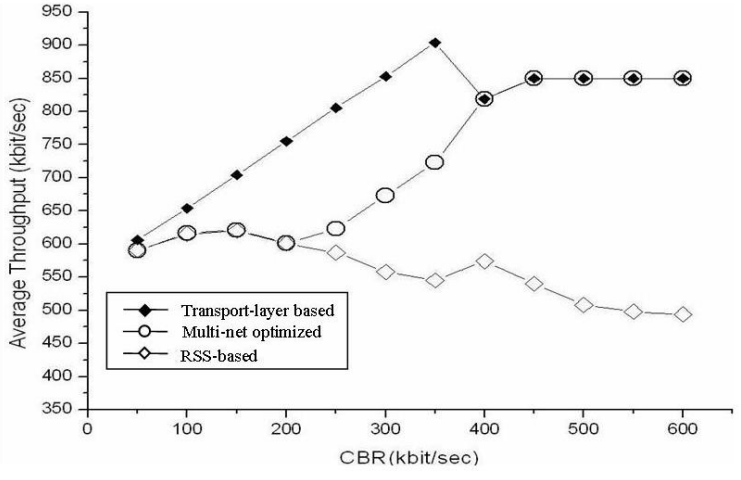


Fig. 3. The total throughput with FTP and CBR traffics

traffic uses WLANs and shows higher throughput than RSS-based and multi-net optimized methods, since in this region, the FTP traffic competes with the CBR traffic in case of the RSS method and multi-net optimized methods. On the other hand, when CBR request is greater than the bandwidth of UMTS cellular networks, the CBR traffic is allocated to WLAN and the FTP traffic is assigned to UMTS cellular networks. Also, mult-net optimized method determine the same allocations. However, the RSS based allocates both traffic to WLANs.

5 Conclusion

This paper proposes a transport-based simultaneous access scheme in integraed WLAN/UMTS mobile networks. Two representative transport protocols, UDP and TCP react differently against delay and loss. WLANs and UMTS cellular networks are dif ferent in not only bandwidth but also in delay and loss. With these different, we try to find transport-layer based traffic allocations to maximize the total transport-layer throughput. To do this, we discuss each transport protocol performance in these two networks and evaluate the proposed algorithm by using NS-2 simulations. The simulation results shows the throughput enhancements.

References

1. Antonio Iera, Antonella Molinaro, Sergio Polito, and Giuseppe Ruggeri, End-to-End QoS Provisioning in 4G with Mobile Hotspots, IEEE Network, vol 29, Sep.-Oct. 2005, pp. 26- 34.
2. Milind M.Buddhikot, Girish Chandranmenon, Seungjae Han, Yui-Wah Lee, Scott Miller, and Luca Salgarelli, Design and Implementation of a WLAN/CDMA2000 Internetworking Architecture, IEEE Commun. Mag., vol 41, Nov. 2003, pp. 90-100.
3. Chunanxioung Guo, Zhihua Guo, Qian Zhang, and Wenwu Zhu, A Seamless and Proactive End-to-End Mobility Solution for Roaming Across Heterogeneous Wireless Networks, IEEE J. Select. Areas Commun., vol. 22, Jun. 2004, pp.834-848.
4. Janise McNair and Fang Zhu, Vertical Handoffs in Fourth-Generation Multinetwork Environments, IEEE Wireless Commnun., vol. 11, Jun. 2004, pp. 8-15.
5. J. Padhaye, V. Firoiu, D. F. Towsley, and J. F. Kurose, Modeling TCP Reno performance: A simple model and its empirical validation, Comput. Commun Rev., vol. 8, pp. 133-145, 2000
6. T.V. Lakshman and Upamanyu Madhow, The Performance of TCP/IP for Networks with High Bandwidth-Delay Products and Random Loss, IEEE/ACM Transactions on Networking, vol. 5, Jun. 1997, pp. 336-350.
7. Fei Hu, Jeeraj K.Sharma, Jim Ziobro, An accurate model for analyzing wireless TCP performance with the coexistence of Non-TCP traffic, Computer Networks, vol 42, 2003, pp. 419-439.
8. UCB/LBNL/VINT, The network simulator-ns-2, http://www.isi.edu/nanam/ns.
9. EURANE: Enhance UMTS Radio Access Networks Extensions for ns-2, http://www.tiwmv. nl/eurane/.

Simplified Transformer Winding Modelling and Parameter Identification Using Particle Swarm Optimiser with Passive Congregation

A. Shintemirov, W.H. Tang, Z. Lu, and Q.H. Wu*,**

Department of Electrical Engineering and Electronics
The University of Liverpool, Liverpool L69 3GJ, U.K.
qhwu@liv.ac.uk

Abstract. The paper presents a simplified mathematical model of disc-type transformer winding for frequency response analysis (FRA) based on traveling wave and multiconductor transmission line theories. The simplified model is applied to the FRA simulation of a transformer winding. In order to identify the distributed parameters of the model, an intelligent learning technique, rooted from particle swarm optimiser with passive congregation (PSOPC) is utilised. Simulations and discussions are presented to explore the proposed optimization approach.

Keywords: Transformer winding mathematical model, particle swarm optimiser with passive congregation.

1 Introduction

Transformer is one of the most expensive elements in a power system. Its condition affects the stability and reliability of the system. One of the well-recommended approaches for transformer winding condition monitoring is the frequency response analysis (FRA). It is performed by the simultaneous measurements of amplitude ratio and phase difference between the input and output signals of a transformer in a wide range of frequency up to several mega hertz using the Fourier transform analysis. In practice, obtained FRA graphs are used for visual comparison with the reference ones for the purpose of fault detections.

One of the approaches of transformer winding modelling is to apply the multiconductor transmission line theory [1][2][3][4], which was developed as an extension of the transmission line theory and firstly employed to electrical machine winding analysis in [5]. Each turn of a winding is modelled as a single transmission line which makes these models complex to operate in the case of describing a winding with a large number of turns.

In this paper a simplified transformer winding model, which is the consensus between physical veracity from one side and comparative simplicity from another side, is presented. This model is applied to winding parameter identification

* Corresponding author: Q H Wu, Tel: +44 1517944535, Fax: +44 1517944540.
** Senior Member IEEE.

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 145–152, 2007.
© Springer-Verlag Berlin Heidelberg 2007

from FRA simulations with PSOPC learning. Simulations and discussions are presented to explore the potentials of the proposed optimization approach.

2 Mathematical Model of Transformer Winding

The essential part of high voltage transformer windings is designed as a continuous disc winding, where each disc is consisted of a number of turns wounded in a radial direction. Continuous disc winding has interdisc connections on internal and external sides of the winding and, therefore, wave propagation along each following disc is in opposite direction with respect to the previous one. For the sake of simplicity it is assumed the one-directional flow of injected signal. The conductor length of each disc of the winding is also assumed to be equal.

In order to describe the wave propagation along each winding disc, a mathematical model of a uniform single layer transformer winding is applied [6][7]. Using the following notations to represent winding electrical parameters per unit of length of conductor, geometrical parameters and elements of a measurement chain:

K	average interturn capacitance;	C	average ground capacitance;
C_{d}	average interdisc capacitance;	C_{b}	bushing capacitance;
L	average inductance;	r	average resistance;
g	average interturn conductance;	G_{d}	average interdisc conductance;
G	average ground conductance;	n	total number of discs;
l	total conductor length of each disc;	a	turn length;
Z_{inp}	impedance of an input measurement cable;	Z_{out}	impedance of an output measurement cable;
u_{inp}	injected multivariable frequency signal,		

the relation between the voltages and currents of a transformer winding in the frequency domain can be presented in the matrix form as follows:

$$\frac{\partial}{\partial x}\begin{bmatrix}\mathbf{U}(s,x)\\ \mathbf{I}(s,x)\end{bmatrix} = \begin{bmatrix}\mathbf{0} & -\mathbf{Z}\\ -\mathbf{Y} & \mathbf{0}\end{bmatrix}\begin{bmatrix}\mathbf{U}(s,x)\\ \mathbf{I}(s,x)\end{bmatrix}, \tag{1}$$

where $\mathbf{Z} = \mathtt{diag}\,(Z_1,\, Z_2, \ldots,\, Z_n)$ and

$$\mathbf{Y} = \begin{bmatrix} \frac{Y_1}{1+a^2 Z_1 Y_{\mathrm{s}1}} & \frac{-Q_{12}}{1+a^2 Z_1 Y_{\mathrm{s}1}} & 0 & \cdots & 0 \\ \frac{-Q_{12}}{1+a^2 Z_2 Y_{\mathrm{s}2}} & \frac{Y_2}{1+a^2 Z_2 Y_{\mathrm{s}2}} & \frac{-Q_{23}}{1+a^2 Z_2 Y_{\mathrm{s}2}} & 0 & \cdots \\ \vdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & 0 & \frac{-Q_{(n-2)(n-1)}}{1+a^2 Z_{n-1} Y_{\mathrm{s}(n-1)}} & \frac{Y_{n-1}}{1+a^2 Z_{n-1} Y_{\mathrm{s}(n-1)}} & \frac{-Q_{(n-1)n}}{1+a^2 Z_{n-1} Y_{\mathrm{s}(n-1)}} \\ 0 & \cdots & 0 & \frac{-Q_{(n-1)n}}{1+a^2 Z_n Y_{\mathrm{s}n}} & \frac{Y_n}{1+a^2 Z_n Y_{\mathrm{s}n}} \end{bmatrix}. \tag{2}$$

The impedances and admittances Z, Y, Y_{s} of each disc and interdisc admittances Q per unit conductor length are defined as follows:

$$Z_i = L_i s + r_i \quad \mathtt{and} \quad Y_{\mathrm{s}i} = K_i s + g_i \quad \text{for} \quad i = 1, \ldots, n; \tag{3}$$

$$Y_i = (C_i + C_{\mathrm{d}(i-1)i} + C_{\mathrm{d}i(i+1)})\, s + (G_i + G_{\mathrm{d}(i-1)i} + G_{\mathrm{d}i(i+1)}),$$
$$Q_{(i-1)i} = C_{\mathrm{d}(i-1)i}\, s + G_{\mathrm{d}(i-1)i}; \quad Q_{i(i+1)} = C_{\mathrm{d}i(i+1)}\, s + G_{\mathrm{d}i(i+1)} \quad \text{for} \quad i = 2, \ldots, n-1; \tag{4}$$

$$Y_1 = (C_1 + C_{\mathrm{d}12})\, s + (G_1 + G_{\mathrm{d}12}); \tag{5}$$

$$Y_n = (C_n + C_{\mathrm{d}(n-1)n})\, s + (G_n + G_{\mathrm{d}(n-1)n}). \tag{6}$$

In the above equations s denotes to the Laplace transform operator and x is a spatial coordinate. Each disc has its own parameters denoted by a corresponding disc number. Thus, notations of $U_1(s,0)$, $I_n(s,l)$ and $U_1(s,0)$, $I_n(s,l)$ denote the voltages and currents at input and output ends of the winding in the frequency domain, *i.e.* $x = 0$ of the first disc and $x = l$ of the n^{th} disc.

The solution of matrix equation (1) for the voltages and currents at the ends of the winding, *i.e.* $x = 0$ and $x = l$, is obtained in the form as below [1]:

$$\begin{bmatrix} \mathbf{U}(s,l) \\ \mathbf{I}(s,l) \end{bmatrix} = \mathbf{\Phi}(l) \begin{bmatrix} \mathbf{U}(s,0) \\ \mathbf{I}(s,0) \end{bmatrix}, \tag{7}$$

where $\mathbf{\Phi}$ is the $2n \times 2n$ chain parameter matrix, which can be calculated in the iterative numerical form using the matrix exponential function [1].

In order to derive the transfer function expressions for FRA, the following boundary equalities are applied [2][5]:

$$U_{i+1}(s,0) = U_i(s,l) \quad \texttt{and} \quad I_{i+1}(s,0) = I_i(s,l) \quad \text{for} \quad i = 1, \ldots, n-1. \tag{8}$$

Consequently, the matrix equation (7) is simplified into the form [3][5]:

$$\begin{bmatrix} U_1(s,0) \\ U_2(s,0) \\ \vdots \\ U_n(s,0) \\ U_n(s,l) \end{bmatrix} = \begin{bmatrix} \\ \mathbf{\Omega}(l) \\ \\ \end{bmatrix} \begin{bmatrix} I_1(s,0) \\ 0 \\ \vdots \\ 0 \\ I_n(s,l) \end{bmatrix}, \tag{9}$$

where $\mathbf{\Omega}(l)$ is a $(n+1) \times (n+1)$ matrix.

The following terminal condition is applied to derive the transfer function of a transformer winding for FRA:

$$U_n(s,l) = Z_{\mathrm{out}} I_n(s,l). \tag{10}$$

With the above terminal conditions the transfer function of a disc transformer winding for a FRA test is calculated as the ratio of $U_n(s,l)$ and $U_1(s,0)$ as the following [3]:

$$H(s) = \frac{\Omega_{(n+1,1)} Z_{\mathrm{out}}}{\Omega_{(1,1)} \left(Z_{\mathrm{out}} - \Omega_{(n+1,n+1)}\right) + \Omega_{(1,n+1)} \Omega_{(n+1,1)}}. \tag{11}$$

The preferred method of engineers is to use the Bode Diagram, which plots the magnitude in decibels (dB) or $20\log_{10}(H(\mathrm{j}\omega))$, where the Laplace transform operator $s = \mathrm{j}\omega$ in the frequency domain.

3 Model-Based Approach with Intelligent Learning Using PSOPC

Particle swarm optimiser (PSO) is a population-based evolutionary algorithm originally developed by Kennedy and Eberhart in 1995, which was inspired by the social behavior of animals such as fish schooling or bird flocking [8][9]. It was extensively used to solve various optimisation problems. However, recent studies of PSO indicates that although the PSO outperforms other evolutionary algorithms in the early iterations, it does not improve the quality of the solutions as the number of generations is increased. Comparative analysis has shown that a novel hybrid PSO with passive congregation introduced in [10] outperforms the standard PSO on multi-model and high dimensional optimisation problems.

3.1 Standard Particle Swarm Optimisation

Being a population-based algorithm the main terms of PSO are *swarm* and *particle* which denotes population and individual respectively. The i_{th} particle at iteration k has the following attributes: a current position in an N-dimensional search space $X_i^k = (x_{i,1}^k, \ldots, x_{i,n}^k, \ldots, x_{i,N}^k)$, where $x_{i,n}^k \in [l_n, u_n]$, $1 \leq n \leq N$, l_n and u_n are the lower and upper bounds for the n_{th} dimension respectively; a current velocity $V_i^k = (v_{1,i}^k, \ldots, v_{n,i}^k, \ldots, v_{N,i}^k)$ which is restricted to a maximum velocity $V_{\max}^k = (v_{\max,1}^k, \ldots, v_{\max,n}^k, \ldots, v_{\max,N}^k)$.

At each iteration, the swarm is updated with the following equations:

$$V_i^{k+1} = wV_i^k + c_1 r_1 (P_i^k - X_i^k) + c_2 r_2 (P_g^k - X_i^k); \quad X_i^{k+1} = X_i^k + V_i^{k+1} \ , \quad (12)$$

where P_i is the best previous position of the i_{th} particle (also known as *pbest*), P_g is the global best position among all the particles in the swarm (also known as *gbest*), r_1 and r_2 are elements from two uniform random sequence on the interval $[0,1]$: $r_1 \sim U(0,1)$; $r_2 \sim U(0,1)$, and w is an inertia weight which is typically chosen in the range of [0,1]. The global exploration is achieved by a larger inertia weight whereas a smaller inertia weight tends to facilitate the local exploration to fine-tune the current search area [11]. The inertia weight w is critical for PSO's convergence behavior, since it provides the balance between global and local exploration abilities and consequently produces a better optimum solution. The acceleration constants c_1 and c_2 control the movements of a particle in a single iteration. The maximum velocity $V_{\max}$ is set to be half of the length of the search space.

3.2 PSO with Passive Congregation (PSOPC)

Kennedy emphasizes the following assumptions as a foundation of PSO model: the autobiographical memory which remembers the best previous position of each individual (*pbest*) in the swarm and the publicised knowledge which is the best solution (*gbest*) currently found by the population [8].

However, from biology point of view, the sharing of information among conspecifics is achieved by employing the publicly available information *gbest*. There

is no information sharing among individuals except that *gbest* gives out the information to the other individuals. Therefore, for the ith particle, the search direction will only be affected by 3 factors: the inertia velocity wV_i^k, the best previous position *pbest*, and the position of global best particle *gbest*. The population is likely to lose diversity and confine the search around local minima[10].

In order to improve the search performance, a passive congregation model is incorporated in the PSO model. Passive congregation is an attraction of an individual to the entire group but without displaying of a social behavior. For instance, in spatially well-defined congregations, such as fish schools, individuals may have low fidelity to the group because the congregations may be composed of individuals with little to no genetic relation to each other. Therefore, in these congregations information may be transferred passively rather than actively [10].

It is known that group members in an aggregation can react without direct detection of incoming signals from the environment, because they can receive necessary information from their neighbors. Individuals need to monitor both environment and their immediate surroundings such as the bearing and speed of their neighbors [12]. Therefore each individual in the aggregation has a multitude of potential information from other group members which may minimise the chance of missed detection and incorrect interpretations [12]. Such information transfer is employed in the model of passive congregation. By implementing aforementioned statements and preserving a simplicity and uniformity of a model, a hybrid PSO with passive congregation (PSOPC) has been developed as the following [10]:

$$V_i^{k+1} = wV_i^k + c_1 r_1 (P_i^k - X_i^k) + c_2 r_2 (P_g^k - X_i^k) + c_3 r_3 (R_i^k - X_i^k); \tag{13}$$

$$X_i^{k+1} = X_i^k + V_i^{k+1}, \tag{14}$$

where R_i is a particle randomly selected from the swarm, c_3 the passive congregation coefficient and r_3 a uniform random sequence in the range (0,1): $r_3 \sim U(0,1)$.

3.3 Fitness Function Used by PSOPC Optimisation

The proposed model-based learning approach is used to search the optimal parameters by minimizing the difference (*i.e.* fitness) between original frequency responses and simulated model outputs with PSOPC. It is achieved by measuring the errors between the original responses and the model outputs. Thereby, for each individual (particle) of a population in PSOPC, its total fitness value is given as below:

$$\min \sum_{j=1}^{M} \|H_{\mathrm{o}}(\omega_j) - H(\omega_j)\|, \tag{15}$$

where $H_{\mathrm{o}}(\omega_i)$ and $H(\omega_i) \in \mathbb{R}^1$ are the original and optimised frequency responses at frequency ω_j, $j = 1, \cdots, M$, where M is the number of samples involved for PSOPC optimisation.

4 Simulations and Discussions

Using the calculated values of the winding parameters of a disc type transformer in [13], which are utilized for illustrative purposes and presented in Table 1, a simulated FRA test is carried out to generate the frequency responses of the simplified winding model. Winding inductances L, resistances r and conductances G are estimated according to [2][4] using the parameters cited in Table 1.

Table 1. Winding Parameters (reproduced from [13])

Parameter	Symbol	Value
Number of discs		18
Turns per disc		10
Conductor wight, mm		6.95
Conductor height, mm		11.2
Average turn lenght, mm		1.4828
Thickness of interturn insulation, mm	d	3
Relative permettivity of interturn insulation	ε_{r}	3.5
Conductor conductance, Si/m	σ	3×10^{-7}
Interturn capacitance, pF/m	K	120
Interdisc capacitance, pF/m	C_{d}	10
Ground capacitance, pF/m	C	15

The following procedures are employed to simulate an actual FRA test:

- Employing the predefined model parameters listed in Table 1, equation (11) is used to compute the frequency responses of the model at frequencies varying from 10Hz to 3MHz.
- The generated frequency responses are recorded as dataset 1, which is employed as training targets for the PSOPC optimisation.

It is assumed that the geometrical and material parameters of the tested winding are known, therefore, only interturn, interdisc and ground capacitances are of interest for optimisation. Based upon dataset 1, the PSOPC technique is utilised to identify a set of model parameters in order to obtain frequency responses close to dataset 1. The generations of PSOPC is set as 80 and its fitness function is defined using equation (15).

In Fig. 1, the original frequency response trend and the calculated frequency response trend using the parameters found by PSOPC are displayed for comparison. It is apparent that, the simulated frequency response curve using the parameters identified by PSOPC learning is very close to the original response. The identified parameters with PSOPC optimisation are $K = 123.2$ pF/m, $C_{\mathrm{d}} = 9.8$pF/m and $C = 14.6$pF/m, which are close to the pre-set values.

Comparative analysis between a PSOPC and a Genetic Algorithm (GA) regarding the parameter optimisation of the proposed transformer winding model has shown that GA is able to find similar results with larger population compared

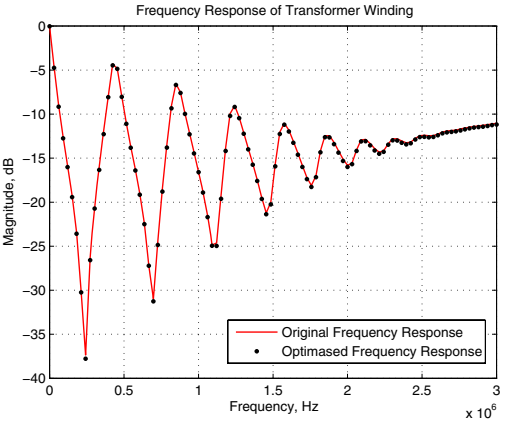


Fig. 1. Comparison between the frequency responses (dataset I) and the model output with PSOPC learning

to the PSOPC's one. Experiments have been carried out with both PSOPC and GA learning, which have involved the same fitness function given by equation (15) and the same data extracted from dataset 1. An advantage of PSOPC is that it has only 2 parameters to adjust, *i.e.* inertia weight and maximum generations, whereas at least 4 parameters in a simple GA, *i.e.* mutation probability, crossover probability, selection probability and maximum generations are needed to be tuned, that makes it particularly attractive from a practitioner's point of view.

5 Conclusion and Further Work

A modified mathematical model of disc-type transformer winding for frequency response analysis is proposed in this paper. During the derivation of the model, each disc is represented by the equations describing traveling wave propagation in a uniform single layer winding. It allows to reduce significantly the order of the model, which is determined only by the number of discs in the modeled winding.

A new approach is proposed to determine the parameters of the proposed transformer winding model with the PSOPC learning. The PSOPC learning has delivered a satisfactory performance during optimisation based upon the original simulated FRA targets. Comparative study with GA has revealed that PSOPC is more efficient for the given optimisation problem. There is a slight difference between the identified parameters and the preset parameters, which is negligible in a practical sense. It can also be deduced that the proposed approach might be further utilised for the winding model parameter identification and trend analysis aiming at winding distortion problems. Additionally, as the method has a simple form and a clear physical meaning, it holds significant potential for the condition assessment of power transformers.

In the current study, only simulations are involved to determine the parameters of the proposed disc type transformer winding model. In general, real FRA datasets should be used for parameter identification and comparison. It may lead to the extension of a number of parameters to be optimised in case of a faulty condition winding, causing nonuniform distribution of parameter values. Therefore, more efficient convergence criterion needs to be utilised. Consequently, further studies will be concentrated on the development of an interpretation procedure upon the measurement data using the proposed model-based learning approach for fault detection aiming at transformer windings.

References

1. Paul, C.R.: Analysis of Multiconductor Transmission Lines. John Wiley & Sons, Inc., New York (1994)
2. Hettiwatte, S., Crosley, P., Wang, Z., A.W.Darwin, Edwards, G.: Simulation of a transformer winding for partial discharge propagation studies. In: IEEE Power Engineering Society Winter Meeting. Volume 22. (January 2002) 1394–1399
3. Jayasinghe, J., Wang, Z., P.N.Jarman, A.W.Darwin: Investigations on sensivity of FRA technique in diagnostic of transformer winding deformations. In: IEEE International Symposium on Electrical Insulation, Indianapolis, USA (September 2004) 496–499
4. Zhang, X., Liang, G., Sun, H., Cui, X.: Calculation of very fast transient overvoltages in power transformer winding. In: Proceedings of the 16th International Zurich Symposium on Electromagnetic Compatibility, Zurich, Switzerland (Febriary 2005) 432–435
5. Guardado, J.L., Cornick, K.J.: A computer model for calculating steep-front surge distribution in mashine windings. IEEE Transactions on Energy Conversion **4**(1) (1989) 95–101
6. Shintemirov, A., Wu, Q.H.: Transfer function of transformer winding for frequency response analysis based on traveling wave theory. In: Proceedings of the International Control Conference (ICC 2006), Glasgow, Scotland (August 2006) 54
7. Rüdenberg, R.: Electrical Shock Waves in Power Systems: Traveling Waves in Lumped and Distributed Circuit Elements. Harvard University Press, Cambridge, Massachusetts (1968)
8. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proceedings of the IEEE International Conference on Neural Networks. Volume 4. (1995) 1942–1948
9. Kennedy, J., Eberhart, R.: Swarm Intelligence. Morgan Kaufmann Publishers (2001)
10. S.He, Wu, Q., Wen, J., Saunders, J., Paton, R.: A particle swarm optimizer with passive congregation. BioSystems **78**(1) 135–147
11. Shi, Y., Eberhart, R.: Parameter selection in particle swarm optimization. In: Evolutionary Programming VII, Lecture Notes in Computer Science. Volume 1447., Springer (1998) 591–600
12. J.K.Parrish, Hamner, W.: Animal Groups in Three Dimensions. Cambridge University Press, Cambridge,UK (1997)
13. Sun, H., Liang, G., Zhang, X., Cui, X.: Analysis of resonance in transformer windings under very fast transient overvoltages. In: Proceedings of the 17th International Zurich Symposium on Electromagnetic Compatibility, Zurich, Switzerland (Febriary 2006) 432–435

A Decentralized Hierarchical Aggregation Scheme Using Fermat Points in Wireless Sensor Networks

Jeongho Son, Jinsuk Pak, Hyunsook Kim, and Kijun Han

{jhson,jspak,hskim}@netopia.knu.ac.kr, kjhan@knu.ac.kr

Abstract. The energy cost for transmission is more expensive than receiving or sensing cost. In this paper, we propose a decentralized aggregation scheme using the Fermat points to save energy for transmitting redundant low-rate data on many-to-one flows in wireless sensor networks. It can reduce the number of transmissions needed to deliver from each source to the sink. Simulation results show that our scheme is better than the GIT (Greedy Incremental Tree) scheme in terms of the number of transmissions and network lifetime.

Keywords: Aggregation, Fermat point, Sensor networks.

1 Introduction

A wireless sensor network consists of many tiny sensor nodes with sensing, processing, and communication capabilities, which communicates in ad-hoc mode and can be used in a variety of applications from defense systems to environmental monitoring [1][2][4].

Wireless sensor networks must minimize the energy consumption since a sensor node has limited battery power [2]. To prolong network lifetime, sensor nodes should use energy more effectively for sensing, transmitting, or processing [9]. The energy cost for transmitting data is much higher than that for receiving, sensing or processing in wireless sensor networks. Therefore, if we can reduce the number of transmissions, we can save energy [$Tx + Rx \times$ *(Average number of neighbors)*] and the network lifetime is prolonged. Data aggregation is one of methods used to reduce the number of transmissions as in-network processing [3]. Different data packets from many sources are mixed into one to reduce the number of transmissions. In-network processing can compress the size of data in order to conserver the transmission energy.

We propose a new aggregation scheme, called the Fermat point Aggregation Scheme (FAS) that data aggregation at Fermat points can minimize the sum of distances from each source to the sink. Our scheme contributes to prolonging the network lifetime by reducing the number of transmissions.

We composed our paper as follows. Related works are described in section 2. Section 3 shows our scheme, and simulation results in section 4 are presented. Finally, we conclude our paper in section 5.

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 153–160, 2007.
© Springer-Verlag Berlin Heidelberg 2007

2 Related Works

Data aggregation is one of methods to save energy in wireless sensor networks by eliminating redundant transmissions from multiple sources to the sink. Data aggregation is defined as the task of merging messages while they are traveling through the sensor network. Reducing the number of messages transmitted by data aggregation in a network can greatly save the amount of energy consumed [3][8][11].

The optimal aggregation, however, is the NP-hard problem of the minimum Steiner problem on graph [2][17]. Therefore, many suboptimal aggregation schemes have been proposed, including the Center at Nearest Source scheme (CNS), the Shortest Path Tree scheme (SPT), and the Greedy Incremental Tree scheme (GIT). The GIT scheme is known to be the best suboptimal solution of them for aggregating data to reduce the number of transmissions. In the GIT scheme, the aggregation tree initially consists of only the shortest path between the sink and the nearest source. And, the tree is expanded in such a way that the next source closest to the current tree is connected to the tree. These schemes require that the sink node should know positions of all nodes and each node should have a global address. They also need the link state information of all nodes to route data [2].

3 Proposed Scheme

We should determine the points to minimize the number of transmissions (hops) from each source to a sink. In densely-deployed networks, we can assume that the hop count from the source to the sink is in proportion to the physical distance. For this reason, we should select aggregation points which minimize total distance from source to the sink so as to save transmission energy.

Assuming that there are one sink (P1) and three sources (P2, P3, and P4) as shown in Fig. 1, then two Fermat points F1 and F2 can minimize the sum of distance from each source and the sink. In our scheme, data from P3 and P4 are first aggregated at F2, and then data from P2 and F2 are again aggregated at F_1 and finally delivered to the sink P.

The procedure to attain the Fermat points in a rectangle is as follows.

(1) Make two equilateral triangles, $\Delta P_1 L P_2$ and $\Delta P_3 R P_4$, on each side P_1P_2 and P_3P_4, respectively, as shown in Fig. 1. We can get vertices L and R, by circularly shifting P_1 and P_3, respectively, by 60° in the counterclockwise direction. The coordinates of $L(x_L,y_L)$ and $R(x_R,y_R)$ can be calculated by

$$\begin{aligned} x_L &= d_{12}\cos(\alpha_1 - \frac{\pi}{3}) + x_1, \quad y_L = d_{12}\sin(\alpha_1 - \frac{\pi}{3}) + y_1 \\ x_R &= d_{34}\cos(\alpha_3 - \frac{\pi}{3}) + x_3, \quad y_R = d_{34}\sin(\alpha_3 - \frac{\pi}{3}) + y_3 \end{aligned} \tag{1}$$

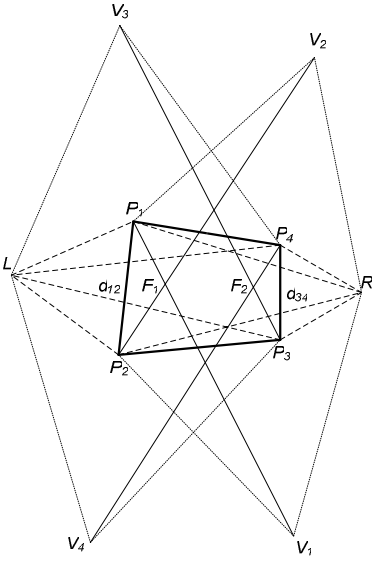


Fig. 1. Hierarchical aggregation for a rectangle

where d_{mn} means the distance from P_m to P_n, and $\alpha_1, \alpha_2, \alpha_3$ and α_4 mean the slope of each side P_{12}, P_{23}, P_{34}, and P_{41}, respectively.

(2) In the same way, we construct three equilateral triangles (ΔP_1V_2R , ΔP_2V_1R , ΔP_3RP_4 and ΔP_3V_4L , ΔP_4V_3L , ΔP_1LP_2) on each side of ΔP_1RP_2 and ΔP_3LP_4. $V_1(x_1', y_1')$, $V_2(x_2', y_2')$, $V_3(x_3', y_3')$ and $V_4(x_4', y_4')$ are by

$$
\begin{aligned}
x_1' &= d_{2R}\cos(\alpha_{2R} - \frac{\pi}{3}) + x_2, \quad y_1' = d_{2R}\sin(\alpha_{2R} - \frac{\pi}{3}) + y_2 \\
x_2' &= d_{R1}\cos(\alpha_{R1} - \frac{\pi}{3}) + x_R, \quad y_2' = d_{31}\sin(\alpha_{R1} - \frac{\pi}{3}) + y_R \\
x_3' &= d_{4L}\cos(\alpha_{4L} - \frac{\pi}{3}) + x_4, \quad y_3' = d_{4L}\sin(\alpha_{4L} - \frac{\pi}{3}) + y_4 \\
x_4' &= d_{L3}\cos(\alpha_{L3} - \frac{\pi}{3}) + x_L, \quad y_4' = d_{L3}\sin(\alpha_{L3} - \frac{\pi}{3}) + y_L
\end{aligned}
\tag{2}
$$

where α_{2R} , α_{R1} , α_{4L} , and α_{L3} mean the slope of the lines P_2R, RP_1, P_4L, and LP_3, respectively.

(3) We can obtain one Fermat point (F_1) at the intersection of two segments V_3-P_3 and V_4-P_4 and another (F_2) at the intersection of two segments V_2-P_2 and V_1-P_1. The coordinates of two Fermat points are given by Equation (3) and (4).

$$x_{F_1} = \frac{y_2 - y_1 + \frac{y_1 - y_1'}{x_1 - x_1'} x_1 - \frac{y_2 - y_2'}{x_2 - x_2'} x_2}{\frac{y_1 - y_1'}{x_1 - x_1'} - \frac{y_2 - y'_2}{x_2 - x_2'}}$$

$$y_{F_1} = \frac{y_1 - y_1'}{x_1 - x_1'} \left(\frac{y_2 - y_1 + \frac{y_1 - y_1'}{x_1 - x_1'} x_1 - \frac{y_2 - y_2'}{x_2 - x_2'} x_2}{\frac{y_1 - y_1'}{x_1 - x_1'} - \frac{y_2 - y'_2}{x_2 - x_2'}} - x_1 \right) + y_1 \tag{3}$$

$$x_{F_2} = \frac{y_4 - y_3 + \frac{y_3 - y_3'}{x_3 - x_3'} x_3 - \frac{y_4 - y_4'}{x_4 - x_4'} x_4'}{\frac{y_3 - y_3'}{x_3 - x_3'} - \frac{y_4 - y_4'}{x_4 - x_4'}}$$

$$y_{F_2} = \frac{y_3 - y_3'}{x_3 - x_3'} \left(\frac{y_4 - y_3 + \frac{y_3 - y_3'}{x_3 - x_3'} x_3 - \frac{y_4 - y_4'}{x_4 - x_4'} x_4'}{\frac{y_3 - y_3'}{x_3 - x_3'} - \frac{y_4 - y_4'}{x_4 - x_4'}} - x_3 \right) + y_3 \tag{4}$$

As mentioned above, many conventional schemes should construct routing trees in a complicated way at a very expensive cost. Our proposed scheme can be easily implemented in a decentralized mode based on Directed diffusion.

For implementation of our scheme, we employ three kinds of control messages: Interest message, exploratory-data message, and reinforcement message. Interest message is created and issued by the sink in order to get information from designate area as a new subscription. Exploratory message is sent by the source as a reply to the interest message issued by the sink. It is used to find the best path from the source to the sink. Reinforcement message is needed for the sink to decide which path should be selected.

In Fig. 2 (a), the sink that is interested in certain events over some area creates an interest message. An interest message contains a description of the desired data, a data rate, an expiration time, target area, and the location of the Fermat point for the target area. The sink floods the interest message through the network. Intermediate nodes store this information in their cache table. The entries, called gradients, in this table are used as backward paths. Each source obtains data from sensor field and makes an exploratory message, and then sends this message to the sink over gradients as illustrated in Fig. 2(b). The node designated near the Fermat point marks flows from every source node, and sends them to the sink using gradient information to help the sink select a path. As shown in Fig. 2(c), the sink that has received messages from each source selects a path to reinforce, and intermediate nodes reinforce the marked path recursively. After whole paths are setup completely, each source sends the sensed data to the sink.

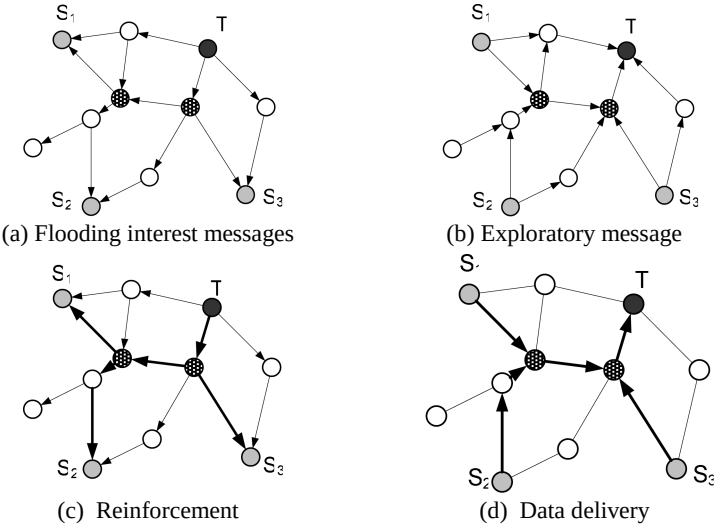


Fig. 2. The procedure for data aggregation using the Fermat points

4 Simulation Results

We evaluate our scheme by comparing with the GIT scheme via a simulation study. For simulation, we assume that stationary sensor nodes are randomly deployed and each sensor node has a transmission range of 15*m* and spends 16*mW* on transmission and *12mW* on receiving. We ignore the idle and computational energy because we focus on only the impact of the number of transmissions on the energy consumption. Each node is initialized with the energy of 1000*mW*, but the sink has unlimited

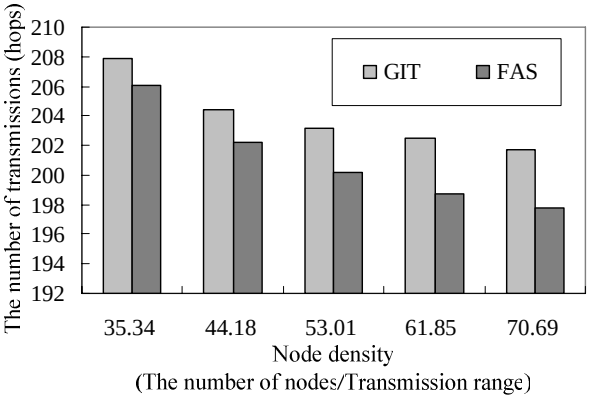
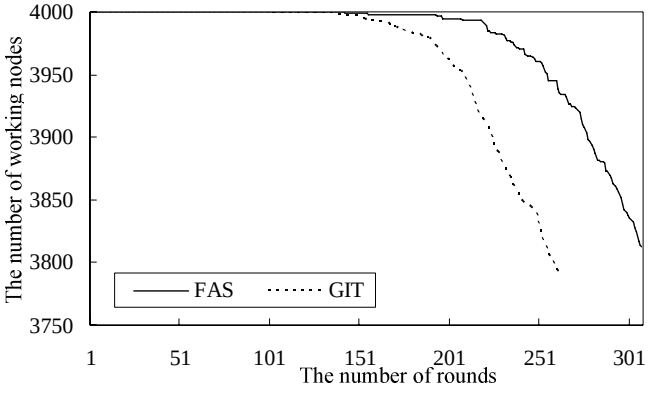


Fig. 3. The number of transmissions when network density is varied

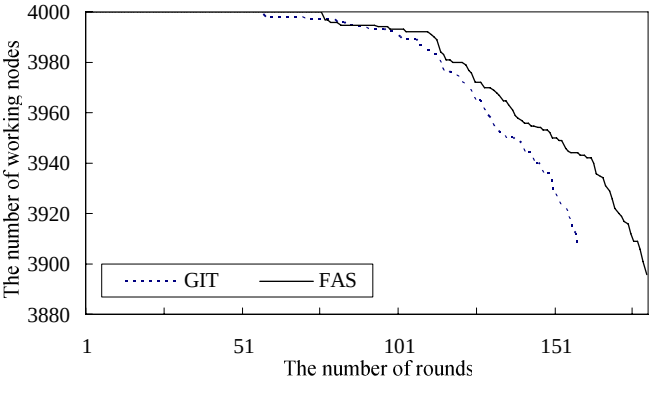
energy. And, we assume that each node knows its location information at least, since each node should determine whether it is near Fermat point or not.

First, we investigate the impact of the network density on network lifetime. To change network density, we vary the number of nodes deployed on the field size of $200m \times 200m$. The task is defined as each source sends its data to the sink at every one minute for 10 minutes. The simulation results show the number of transmissions (hops) accumulated during the task. The sink is located at the most left-bottom, that is, at the coordinate of (0, 0). As shown in Fig. 3, the number of transmissions required in the network is inversely proportional to the network density.

We can see that the proposed scheme offers a more desirable performance as the network density becomes higher. This indicates that the network density is a very important factor in our scheme. We recommend that the proposed scheme use in dense networks since we can construct a straight path from sources to the sink



(a) When the sink is located at the center of network



(b) When the sink is located at the most left-bottom

Fig. 4. Network lifetime

through the Fermat point in densely deployed networks. In sparse networks, the aggregation node may not be selected near the Fermat point calculated by the sink. In other words, the path from each source to the sink through the Fermat points can be longer in sparsely deployed network. In the GIT scheme, one path consists of 3 segments while the FAS scheme yields 5 segments per path. It means that each segment may contain more hops in FAS than in GIT.

Fig. 4 shows the impact of the location of the sink on the network lifetime. The number of nodes is 4000, and each node has transmission range of 15*m*. Fig. 4 (a) shows variation of the residual energy of the network and the number of working nodes as the time goes on when the sink is located at the center of network. Fig. 4 (b) shows the performance when the sink is located at the most left-bottom corner of network.

When the sink is located at the center of networks, the network dies at the 308th round that means there are no paths between sources and the sink. However, when the sink is located at the corner, the network lifetime becomes shortened (becomes dead at the 182nd round). This is because there are more neighbors at the center of field, and thus more available paths can be established near the sink.

In Fig. 5, the task is defined as each source sends its data to the sink at every one minute for 10 minutes. The simulation results show the number of transmissions (hops) accumulated during the task, as the network size is varied between 150 and 230 by increments of 20 with fixed density (2250, 2890, 3610, 4410, and 5290 respectively). The differences between the GIT scheme and FAS increase as the network size increases. Consequently, we can say that the FAS can be more effective in the high density network with a large size field.

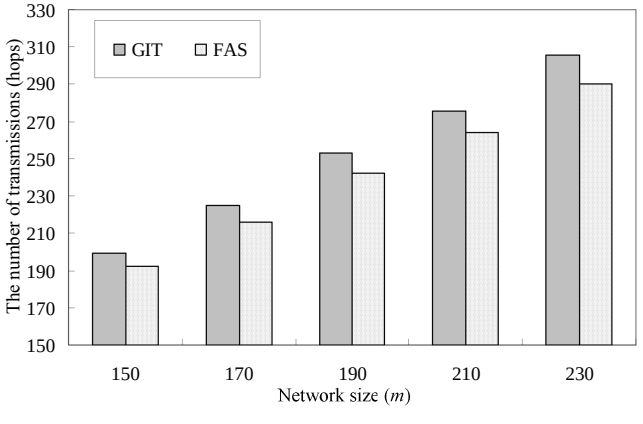


Fig. 5. The network lifetime versus the network field size

5 Conclusion

We have proposed an aggregation scheme operating in a decentralized mode for easy application to wireless sensor networks. The proposed scheme is an optimal scheme

which employs a kind of mathematical approach. The number of transmissions from the source to the sink is in proportion to the physical distance in densely-deployed networks. Therefore, we used Fermat points to minimize the number of transmissions from each source to the sink. The simulation results show that our scheme offers a better performance than the GIT scheme in terms of the number of transmissions. Our scheme is more effective in densely deployed networks covering a wide area.

References

1. Tatiana Bokareva, Nirupama Bulusu, Sanjay Jha, "A Performance Comparison of Data Dissemination Protocols for Wireless Sensor Networks, Global Telecommunications Conference Workshops," *GlobeCom Workshops 2004*, pp. 85-89, 2004.
2. Bhaskar Krishnamachari, Deborah Estrin, Stephen Wicker, "The Impact of Data Aggregation in Wireless Sensor Networks," *Preceedings of the 22nd International Conference on Distributed Computing Systems Workshops (ICDCSW'02)*, pp. 575-578, 2002.
3. Chalermek Intanagonwiwat, Ramesh Govindan, Deborah Estrin, John Heidemann, "Directed Diffusion for Wireless Sensor Networking", *Networking, IEEE/ACM Transactions*, vol. 11, pp. 2-16, 2003.
4. Chalermek Intanagonwiwat, Ramesh Govindan and Deborah Estrin, "Directed diffusion: a scalable and robust communication paradigm for sensor networks," *Proceedings of the 6th annual international conference on Mobile computing and networking (MobiCom'02)*, pp. 56-67, 2002.
5. http://mathsforeurope.digibel.be/fermat.htm, (2006)
6. http://mathworld.wolfram.com/FermatPoints.html, (2006)
7. Qing Cao, Tian He, Tarek Abdelzaher, "uCast: Unified Connectionless Multicast for Energy Efficient Content Distribution in Sensor Networks," *IEEE Transactions on Parallel and Distributed Systems*, 2006.
8. Wolf, T.; Choi, S.Y., "Aggregated hierarchical multicast-a many-to-many communication paradigm using programmable networks,", *IEEE Transactions on Systems, Man and Cybernetics*, vol. 33, issue 3, pp. 358-369, 2003.
9. Landsiedel, O.; Wehrle, K.; Gotz, S., "Accurate Prediction of Power Consumption in Sensor Networks, Embedded Networked Sensors," 2005. *EmNetS-II*, pp. 37-44, 2005.
10. http://user.chol.net/~badang25/fermat/fermat04.htm (2006)
11. Hong-Hsu Yen; Lin, F.Y.-S.; Shu-Ping Lin; "Efficient data-centric routing in wireless sensor networks," *IEEE International Conference*, vol. 5, pp. 3025-3029, 2005.
12. Roedig, U.; Barroso, A.; Sreenan, C.J. "Determination of aggregation points in wireless sensor networks," *Euromicro Conference*, pp. 503-510, 2004.

A Gateway Access-Point Selection Problem and Traffic Balancing in Wireless Mesh Networks

A. Cagatay Talay*

Department of Computer Engineering, Istanbul Technical University,
34469 Istanbul, Turkey
ctalay@itu.edu.tr

Abstract. A wireless mesh network (WMN) composed of multiple access-points (APs) that communicate mutually using radio transmissions, and all the traffics to/from the Internet are aggregated and go through the limited number of gateway APs. The meshed topology provides good reliability, market coverage, and scalability, as well as low upfront investment. However, due to the nature of routing algorithm the traffic load for a access point may be extremely heavy during particular periods while the other access points are in very light load. Consequently, the overall performance of the network are poor even the total traffic load is far below the system capacity. The performance can be improved dramatically by Traffic Balancing. Strategically placing and connecting the gateways to the wired backbone is critical to the management and efficient operation of a WMN. In this paper, we address the problem of gateway access-points placement, consisting in placing a minimum number of gateways such that quality-of-service (QoS) requirements are satisfied. We propose a genetic algorithm that consistently preserves QoS requirements. We evaluate the performance of our algorithm using both analysis and simulation, and show that it outperforms other alternative schemes by comparing the number of gateways placed in different scenarios.

1 Introduction

Wireless is well established for narrowband access systems, but its use for broadband access is relatively new. Wireless mesh architecture is a first step towards providing high bandwidth network coverage. Mesh architecture sustains signal strength by breaking long distances into a series of shorter hops. Intermediate nodes not only boost the signal, but cooperatively make forwarding decisions based on their knowledge of the network. Such architecture provides high network coverage, spectral efficiency, and economic advantage. Recently, interesting commercial applications of wireless mesh networks (WMNs) have emerged. One example of such applications is "community wireless networks" [1], [2]. Several vendors have recently offered WMN products. Some of the most experienced in the business are Nortel [3], Tropos Networks [4], and BelAir Networks [5]. WMNs have a relatively stable

* The author is partially supported by TUBITAK (The Scientific and Technical Research Council of Turkey).

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 161–168, 2007.
© Springer-Verlag Berlin Heidelberg 2007

topology except for occasional node failures or additions. Practically all the traffic flows either to or from a gateway, as opposed to ad hoc networks where the traffic flows between arbitrary pairs of nodes. Gateways would be connected directly to the fixed network, and therefore constitute traffic sinks and sources to WMNs. Therefore, strategically placing and connecting the gateways to the wired backbone is critical to the management and efficient operation of a WMN.

The analysis of WMN scalability is based on the following scaling relationships: traffic increases with the number of nodes, and traffic also increases with the distance over which each node wishes to communicate (i.e., due to packet forwarding). In [6], Li *et al.* showed that , the capacity available to each node (i.e., the rate at which packets are originated), is bounded by $\lambda < \frac{C/n}{\bar{L}/r}$ where C is the total one-hop capacity of the network, n is the number of nodes, $\bar{L}$ is the expected path length, and r is the fixed radio transmission range such that $\bar{L}/r$ is the minimum number of hops to deliver packets.

The above inequality shows that as the expected path length increases, the bandwidth available for each node to originate packets decreases. Therefore, the network scales better when the traffic pattern is *local*. That is, each node sends only to nearby gateways within a fixed radius, independent of the network size. The expected path length clearly remains constant as the network size grows. Hence, for optimal performance, the WMN should be divided into disjoint clusters, covering all nodes in the network. Within each cluster, the clusterhead would serve as the gateway, connected to the wired backbone.

A tree-based routing scheme would easily allow flows aggregation and would minimize overhead, ensuring an optimal utilization of bandwidth [7]. Hence, a spanning tree rooted at the gateway can be used for traffic forwarding. Each node is mainly associated to one tree, and would attach to another tree as an alternative route in case of path failure.

For operational considerations, the gateway placement problem should take into account the quality-of-service (QoS) requirements such as delay and bandwidth. In a multihop network, significant delay occurs at each hop due to contention for the wireless channel, packets processing, and queueing delay. The delay is therefore a function of the number of communication hops between the source and the gateway. The delay constraint is translated into an upper bound R on the *cluster radius*, or a maximum depth R of the spanning tree rooted at the gateway.

Bandwidth requirements are of two forms. First, the total traffic inside each cluster is bounded by the capacity of the gateway, based on its connectivity to the internet and its processing speed. This requirement is translated into an upper bound on the *cluster size* S , assuming each AP generates an equal amount of one unit of traffic.

Guaranteeing a throughput for individual flows in a multihop wireless network is more challenging. For convenience, we assume a multichannel WMN where interfering wireless links operate on different channels, enabling multiple parallel transmissions. The bottleneck on throughput is therefore reduced to the load of congested intermediate wireless links. Since traffic is aggregated and forwarded by intermediate APs, we refer to the load on individual wireless links as *relay load* L in

unit of traffic. Therefore, the throughput requirement is translated to an upper bound on the *relay load* equal to the capacity of individual wireless links in unit of traffic.

In this paper, we address the problem of gateway placement in a WMN, aiming at placing a minimum number of gateways, while ensuring the QoS requirements discussed above. We present a Genetic algorithm to divide the WMN into clusters of bounded *radius* under *relay load* and *cluster size* constraints.

The rest of this paper is organized as follows. Section 2 presents an overview of related works. Section 3 describes the network model, presents the gateway placement problem. Section 4 presents the proposed algorithm. Experimental analyses and comparison to alternative approaches are performed in Section 5. Section 6 concludes this paper.

2 Related Work

Our work inherits two major concepts from the literature: the capacity facility location problem (CFLP), and clustering and hierarchical routing in ad hoc networks.

The gateway placement problem could be considered as an instance of the more general CFLP problem which has been studied in the fields of operations research and approximation algorithms.

There have been numerous studies on designing hierarchical routing architectures for ad hoc networks. Routing, based on a connected dominating set (CDS) forming a spine to relay routing information and data packets, is a typical technique in MANETs [8]–[10]. The approximation algorithms developed to solve the CDS problem are not suitable in our context: simply relaxing the problem of connecting clusterheads leads to nonoptimal solutions. In addition, the proposed schemes are concerned with one-hop clustering, which defeats the purpose of WMN.

Other works have proposed -hop clustering algorithms [11], [12] but none of them satisfy all the requirements of our clustering problem and rarely present a guarantee in comparison to the optimal performance.

To date and to the best of our knowledge, very few schemes have been proposed to integrate the WMN with the wired backbone.

In [13], Wong *et al.* addressed the gateway placement problem in two separate settings: either minimizing communication delay or minimizing communication cost. For each setting, they propose different statistically tuned heuristics, using the same strategy: at each step they decide which of the candidate gateways will be eliminated from further consideration. QoS constraints in terms of bounds on the *relay load* and *cluster size* are not considered. Furthermore, the proposed approximation algorithm gives no guarantee on the optimality of the solution. The additional QoS constraints considered in this paper make the problem more challenging.

In [14], Chandra *et al.* addressed the problem of gateway placement, aiming at minimizing the number of gateways, while guaranteeing AP's bandwidth requirements. They considered the problem as an instance of the network flow problem, allowing multipath routing. However, when constraints on communication path length are imposed, the proposed greedy heuristics leads to nonoptimal solutions, and hence no guarantee on performance. In addition, the iterative greedy approach makes the load of the gateways unbalanced, since gateways are placed whenever

others are fully served. Finally, a clustered view of the WMN is not considered, making the design less suitable to our context.

The most relevant work to ours is the one in [15]. Bejeranos successfully adopts a clustered view of the WMN and used a spanning tree rooted at each clusterhead (i.e., gateway) for message delivery. Bejerano breaks the problem of clustering and ensuring QoS into two subproblems. The first one seeks to find a minimal number of disjoint clusters containing all the nodes subject to an upper bound on *clusters' radius*. The second one considers placing a spanning tree in each cluster, and clusters that violate the *relay load* or *cluster size* constraints are further subdivided. In this paper, we consider the combined problem, where the spanning tree and cluster coverage evolve in parallel as long as QoS requirements are satisfied. We show that the number of gateways required by our algorithm, subject to the same QoS requirements, is reduced by almost 1/2 in some cases, thus leading to a significant saving in deployment cost.

3 Network Model, Problem Description and Proposed Technique

We consider the problem of gateway placement in the context of WMN. A WMN is represented by an undirected graph $G(V,E)$, called a connectivity graph. Each node $v \in E$ represents an Access Point (AP) with a circular transmission range of 1 unit. The neighborhood of v, denoted by $N(v)$, is the set of nodes residing in its transmission range. A bidirectional wireless link exists between v and every neighbor $u \in N(v)$ and is represented by an edge $u,v \in E$. The number of neighbors of a vertex v is called the degree of v, denoted by $\delta(v)$. The maximum degree in a graph G is called the graph degree $\Delta(G) = \Delta$.

The distance, denoted by $d(u,v)$, between two nodes u and v is the minimum number of hops between them. The radius of a node v in $G(V,E)$ is the maximum distance between v and any other node. The radius of G is hence defined as the minimum radius in the graph. On the other hand, the diameter G of is the maximum distance between two arbitrary nodes (or the maximum radius).

In this paper, we address the efficient integration of theWMN with the wired network for Internet access, while ensuring QoS requirements. This consists in logically dividing the WMN into a set of disjoint clusters, covering all the nodes in the network. In each cluster, a node would serve as a gateway, connected directly to the wired network, and serving the nodes inside the cluster.

In each cluster, a spanning tree rooted at the gateway is used for traffic aggregation and forwarding. Each node is mainly associated to one tree, and would attach to another tree as an alternative route in case of path failure.

For operational reasons, the gateway placement or clustering problem is subject to QoS constraints. As discussed earlier, the QoS constraints are translated into the following: an upper bound R on the cluster radius, an upper bound S on the cluster size, and an upper bound L on relay traffic. The gateway placement problem therefore consists in logically dividing the WMN into a minimum number of disjoint clusters that cover all nodes and satisfy all three QoS constraints.

3.1 Weighted Clustering Algorithm

The Weighted Clustering Algorithm (WCA) was originally proposed by M. Chatterjee et al.[16]. It takes four factors into consideration and makes the selection of clusterhead and maintenance of cluster more reasonable. As is shown in equation (1), the four factors are node degree difference, distance summation to all its neighboring nodes, velocity and remaining battery power respectively. And their corresponding weights are w_1 to w_4.Besides, it converts the clustering problem into an optimization problem and an objective function is formulated.

$$W_i = w_1\Delta_i + w_2 D_i + w_3 V_i + w_4 E_i \tag{1}$$

However, only those nodes whose neighbor number is less than a fixed threshold value can be selected as a clusterhead in WCA. It is not very desirable in the practical application. For example, many well-connected nodes whose neighbor number is larger than the fixed threshold might be a good candidate as well. Besides, its energy model is too simple. It treats the clusterhead and the normal nodes equally and its remaining power is a linear function of time, which is also not very desirable. So, we proposed an improved clustering algorithm as follows.

3.2 Proposed Algorithm

The selection of clusterheads set, which is also called dominant set in Graphic Theory, is a NP-hard problem. Therefore, it is very difficult to find a global optimum. So, we can take a further step to use the computational intelligence methods, such as Genetic Algorithm (GA) or Simulated Annealing (SA), to optimize the objective function.

The steps of our algorithm are as follows. And it usually takes 5 to 10 iterations to convergence. So, we can say that it converges very fast.

Step 1: As for N nodes, randomly generate L integer arrangements in the range of [1, N].

Step 2: Using random arrangements and the clustering principle of WCA, derive L sets of clusterheads and compute their corresponding $\sum w_{iold}$.

Step 3: Using the Roulette Wheel Selection and Elitism in GA, select L sets of clusterheads which are better, and replace the original ones.

Step 4: As for each of the L sets of clusterheads, perform the crossover operator and derive the new L sets of clusterheads and their $\sum w_{inew}$.

Step 5: According to the Metropolis “accept or reject” criteria in SA, decide whether to take the one from L sets of clusterheads $\sum w_{iold}$ or in $\sum w_{inew}$. And the new L sets of clusterheads in the next generation are obtained.

Step 6: Repeat Step 3 to 5 until it converges or a certain number of iteration is reached. And in our simulation, it usually takes 5 to 10 iterations to converge.

Then the global optimal or sub-optimal solution min ($\sum w_{inew}$) (i=1, 2…L) is obtained and their corresponding set of clusterheads is known.

In Step 2, we make L random arrangements in order to reduce the randomness in the clustering process, because there is much difference in the set of clusterheads (or dominant set) for different nodes arrangements. As for the Roulette Wheel Selection in Step 3, we do not take the traditional selection probability, but $P_i = \frac{e^{-\sum w_i}}{\sum_{i=1}^{L}(e^{-\sum w_i})}$. In that case, the set of clusterheads whose $\sum w_i$ is smaller will have more chance to be selected. Besides, to overcome the randomness in the probability problem, we preserve the best set of $\sum w_{iold}$ directly to the in $\sum w_{inew}$ Elitism.

To further reduce the randomness and increase the probability that the global solution may occur, we perform M pairs of crossovers as for each of L random arranged integers (i.e. mobile nodes). And the new L sets of clusterheads and their $\sum w_{inew}$ (i=1,2,…L) are derived. In Step 5, we make the "accept or reject" decision according to the Metropolis criteria. If $\sum w_{inew} \le \sum w_{iold}$ then we accept $\sum w_{inew}$ directly. If $\sum w_{inew} > \sum w_{iold}$, we do not reject it directly, but accept it with some probability. In other words, if $e^{\frac{\sum w_{inew} - \sum w_{iold}}{\alpha T}}$ is larger than a randomly generated number in the range of (0,1), which shows that $\sum w_{inew}$ and $\sum w_{iold}$ may be very close to each other, we will still take it. Or else, we will reject the one in $\sum w_{inew}$ and take its counterpart in $\sum w_{iold}$. Besides, we let $T = \alpha T$ (α is a constant between 0 and 1 and we normally take 0.9) after each iteration, so that $\sum w_{inew}$ and $\sum w_{iold}$ must be closer is $\sum w_{inew}$ is to be accepted. In this way, our algorithm will not be trapped in the local optima and the premature effect can be avoided. In other words, the diversity of searching space can be ensured and it is similar to the mutation operator in GA.

4 Performance Evaluation

We set our simulation environment as follows. There are N nodes randomly placed within a range of 100 by 100 m^2, whose transmission range varies from 15m to 50m. A Random Waypoint mobility model is adopted here. And we take M =1, L = 10, α = 0.9 and ε = 0.01 in our simulations. As is shown in figure 1, we simulate N nodes whose transmission range varies from 15m to 50m. We can conclude that:

(1) The average cluster number (ACN) decreases as the transmission range increases.
(2) As for a smaller transmission range, the average number of cluster differs greatly for various N. But when the transmission range is about 50m, one node can almost cover the entire network. So it only takes 3 to 5 clusters to cover all the N nodes.

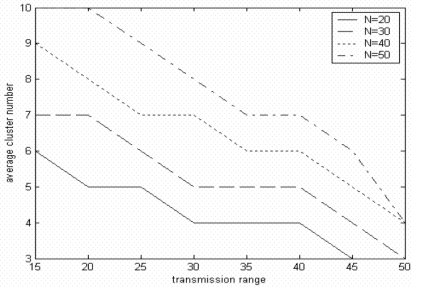


Fig. 1. ACN under various transmission range

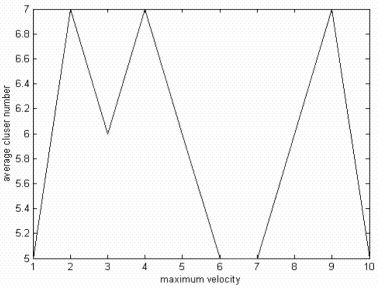
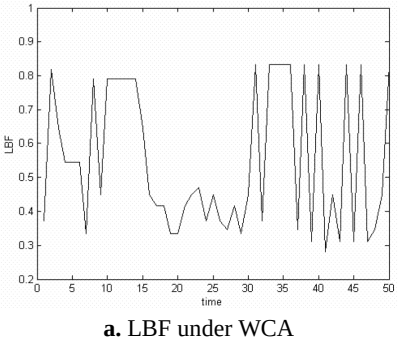
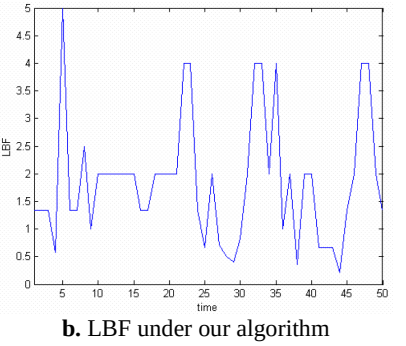


Fig. 2. ACN under various maximum velocities

Besides, we do the same research under various velocities. Taking N=R=30 as an example, we can draw the conclusion from figure 2 that: the average number of cluster varies randomly between 5 and 7, and it is not related with the velocity. In fact, it matches with the practical situation too. For example, when one node with large velocity moves out of a cluster, it is highly possible that some other node gets into the same cluster. Or some of the nodes might move toward the same direction, which results in a relatively slow velocity and a stable cluster too. We take the same definition of load-balancing factor (LBF) as is defined in [16]. The larger LBF is, the better the load is balanced. Taking N=20, M=4 as an example. The ideal case is that there are 4 clusters and each clusterhead has a degree of 4, i.e. $n_c = x_i = 4$. Then, $\mu = (20-4)/4 = 4$. So LBF is infinite, which shows that the load is perfectly balanced. For simplicity, we do not consider the factor of network lifetime here. So we set the simulation parameters as follows. (X,Y)=[100,100], N=20, R=30, M=4, maximum velocity $V_{\max} = 5$ and $w_1 = 0.7, w_2 = 0.2, w_3 = 0.1, w_4 = 0$. It should be noted that we make N_i as our primary focus of attention $w_1 = 0.7$, because it represents the matching degree of the practical case and ideal case directly. Figure 3 shows the LBF



a. LBF under WCA



b. LBF under our algorithm

Fig. 3. Load Balancing Factor

distribution under WCA, and our algorithm. From figure 3 we can see that: our improved clustering algorithm is better. Besides, the WCA will become useless under densely deployed networks while our algorithm still works well. And their average values are 0.38, and 1.86 respectively.

5 Conclusion

In this paper, we elaborated on the importance of clustering for the efficient operation of WMNs. We proposed a novel Genetic Algorithm and Simulated Annealing based algorithm for clustering the WMN while ensuring relay load and cluster size constraints. Some performance comparison is made in the aspect of average cluster number, topology stability, load-balancing and network lifetime. The simulation results show that our algorithm have a better performance on average.

References

1. Seattle Wireless. [Online]. Available: http://www.seattlewireless.net
2. Bay Area Wireless Users Group. [Online]. Available: http://www.bawug.org
3. Nortel. [Online]. Available: http://www.nortel.com
4. Tropos Networks. [Online]. Available: http://www.tropos.com
5. BelAir Networks. [Online].Available: http://www.belairnetworks.com
6. J. Li, C. Blake, D. De Couto, H. Imm, and L. Morris, “Capacity of ad hoc wireless networks,” in Proc. Int. Conf. Mobile Comput. Netw., 2001.
7. P. Hsiao, A. Hwang, H. Kung, and D. Vlah, “Load balancing routing for wireless access networks,” in Proc. IEEE INFOCOM, 1999.
8. V. Bharghavan and B. Das, “Routing in ad hoc networks using minimum connected dominating sets,” in Proc. Int. Conf. Commun., Jun. 1997, pp. 376–380.
9. J.Wu and H. Li, “On calculating connected dominating set for efficient routing in ad hoc wireless networks,” in Proc. Workshop on Discrete Algorithms and Methods for Mobile Comput. Commun., 1999.
10. Y. Chen and A. Liestman, “Approximating minimum size weakly-connected dominating sets for clustering mobile ad hoc networks,” in Proc. ACM Int. Symp. Mobile Ad Hoc Netw. Comput., Jun. 2002.
11. S. Banerjee and S. Khuller, “A clustering scheme for hierarchical control in multi-hop wireless networks,” in Proc. INFOCOM, 2001, pp. 1028–1037.
12. A. Antis, R. Prakash, T. Vuong, and D. Huynh, “Max-min d-cluster formation in wireless ad hoc networks,” in Proc. IEEE INFOCOM, 2000, pp. 32–41.
13. J. Wong et al., “Gateway placement for latency and energy efficient data aggregation,” in Proc. IEEE Int. Conf. Local Comput. Netw., 2004, pp. 490–497.
14. R. Chandra, L. Qiu, K. Jain, and M. Mahdian, “Optimizing the placement of integration points in multi-hop wireless networks,” in Proc. IEEE ICNP, 2004.
15. Y. Bejerano, “Efficient integration of multihop wireless and wired networks with QoS constraints,” IEEE/ACM Trans. Netw., vol. 12, no. 6, pp. 1064–1078, 2004.
16. M. Chatterjee, S.K. Das and D. Turgut. An On-Demand Weighted Clustering Algorithm (WCA) for Ad hoc Networks. Proceedings of IEEE GLOBECOM 2000, San Francisco, November 2000, pp.1697-1701.

A Genetic Programming Approach for Bankruptcy Prediction Using a Highly Unbalanced Database

Eva Alfaro-Cid, Ken Sharman, and Anna Esparcia-Alcázar

Instituto Tecnológico de Informática, Universidad Politécnica de Valencia,
Camino de Vera s/n, 46022 Valencia, Spain

Abstract. In this paper we present the application of a genetic programming algorithm to the problem of bankruptcy prediction. To carry out the research we have used a database of Spanish companies. The database has two important drawbacks: the number of bankrupt companies is very small when compared with the number of healthy ones (*unbalanced data*) and a considerable number of companies have missing data. For comparison purposes we have solved the same problem using a support vector machine. Genetic programming has achieved very satisfactory results, improving those obtained with the support vector machine.

1 Introduction

Bankruptcy prediction is a very important economic issue. It is of great significance for stakeholders as well as creditors, banks, investors, to be able to predict accurately the financial distress of a company. Given its relevance in real life, it has been a major topic in the economic literature. Many researchers have worked on this topic during the last decades; however, there is no generally accepted prediction model.

According to [7], a survey reviewing journal papers on the field in the period 1932-1994, the most popular methods for building quantitative models for bankruptcy prediction have been discriminant analysis [1] and logit analysis [16]. Since the 90s there has been an increasing interest on the application of methods originating from the field of artificial intelligence, mainly neural networks [14].

However, other methods from the artificial intelligence field, such as evolutionary computation have been scarcely used for the bankruptcy prediction problem. After an extensive (but not exhaustive) review of the literature, only a few papers could be found that applied evolutionary methods to the bankruptcy prediction problem.

Some authors have used genetic algorithms (GAs), either on its own [11], [18], [20] or in a hybrid method with a neural network [4] for the insolvency prediction problem. However, most of the approaches from the evolutionary computation field use genetic programming (GP). The ability of GP to build functions make this algorithm more appropriate to the problem at hand than GA. In the literature we can find a couple of hybrid approaches that combine GP with another

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 169–178, 2007.
© Springer-Verlag Berlin Heidelberg 2007

method, such as rough sets [15] and neural networks [19]. Some authors have used GP on its own. In [21], the authors have used linear GP and have compared its performance to support vector machines and neural networks. In [13], the authors have used GP to predict bankruptcy on a database of Norwegian companies and in [17] the GP has been used for the prediction of insolvency in non-life insurance companies, a particular case. Finally, grammatical evolution, a form of grammar-based genetic programming, has been used in [5] to solve several financial problems.

One important advantage of the GP approach to bankruptcy prediction is that it yields the rules relating the measured data to the likelihood of becoming bankrupt. Thus a financial analyst can see what variables and functions thereof are important in predicting bankruptcy.

Our approach differs from previous GP applications in the characteristics of the database we are using. Our database comprises data from Spanish companies from different industrial sectors. The database has two drawbacks: firstly, it is highly unbalanced (only 5-6% of the companies go bankrupt) and, secondly, some data are missing. Although this complicates the classification, it is an accurate reflection of the real world, where few companies go bankrupt in proportion and it is difficult to obtain all the relevant data from companies.

For comparison we have also analyzed the data using a support vector machine (SVM) classifier, and our results demonstrate that our proposed GP technique gives improved performance over the SVM.

2 Financial Data

The work presented in the paper uses a database supplied by the Department of Finance and Accounting of the Universidad de Granada, Spain.

The database consist of a 2859×31 matrix comprising data from 484 Spanish companies from the year 1998 to the year 2003[1]. Each row of the matrix holds the data referent to a company during one year. The database includes not only financial data such as solvency, profit margin or income yield capacity, but also general information such as company size, age or number of partners. These variables are the inputs to the classifier. The desired output of the classifier is the variable that states if the company was bankrupt in 2003 or not.

In this work we have used the data from years 1999 and 2000 to predict bankruptcy in the year 2003, that is 4 and 3 years in advance, respectively.

All variables can take values from different numerical ranges. Some of them take real values, others take real positive values, others take integer values and, finally there are four boolean variables that indicate if the company has been audited, if there was any delay in presenting the accountancy, if the company is linked to a group or if the company has been suffering losses. Therefore, as the numerical range the variables can take varies a lot, all the data have been

[1] The number of rows in the data matrix should be 2904, that is 484×6, but some companies don't have available data for all the years.

normalized between 0 and 1 (in the case of the data being integer, boolean or real positive) or between -1 and 1 otherwise.

One of the problems with this database is that some of the data are missing. Specifically, around 16% of the companies in the database have one or more data values missing. To handle this we have substituted the missing data for the minimum value that variable can take. After the normalization the value will be set to 0 or -1 depending whether the variable can take negative values or not.

2.1 Training and Testing Sets

In order to apply GP to the prediction problem the data sets have been divided into two groups: the training and testing sets, which have been selected randomly. Given that the data base is highly unbalanced (only 5-6% of the companies went bankrupt), this ratio needs to be reflected in the choice of the training set.

The number of companies with available data varies slightly each year. In year 1999 there are data available from 467 companies (27 bankrupt vs. 440 healthy) and in year 2000 there are data available from 479 companies (30 bankrupt vs. 449 healthy). We have kept constant the number of companies in the training set, thus the number of companies in the testing set varies from year to year.

The division of the data into the training and testing sets has been done as follows. The training set consist of 160 companies (10 bankrupt and 150 healthy). The test set for year 1999 consists of 307 companies (17 bankrupt and 290 healthy) and the test set for year 2000 consists of 319 companies (20 bankrupt and 299 healthy).

3 Genetic Programming and Prediction

In this section we briefly describe the GP framework that we have used for representing systems for bankruptcy prediction. Basically, the GP algorithm must find a structure (a function) which can, once supplied with the relevant data from the company, decide if this company is heading for bankruptcy or not. In short, it is a 2-class classification problem for GP to solve. One of the classes consists of the companies that will go bankrupt and the other consists of the healthy ones. For further information on classification using GP see references [8,12].

3.1 Function and Terminal Sets

Prior to creating a GP environment the designer must define which functions (internal tree nodes) and terminals (leaf branches) are relevant for the problem to solve. This choice defines the search space for the problem in question.

The terminal set consists of 30 company data. These data are presented to the classifier as a vector and, in order to simplify the notation, they have been called $x_0, x_1 ... x_{29}$. The evolution process will decide which of these data are relevant for the solution of the problem.

Table 1. Function and terminal set

Nodes	No. arguments	Description
$\mathcal{R}$	0	random constant
$x_0 \ldots x_{29}$	0	company data
$cos,\ log,\ exp$	1	cosine, logarithm, exponential
$+, -, *, /$	2	arithmetic operators
IfLTE	4	if $arg_1 \leq arg_2$ then arg_3 else arg_4

Table 1 shows the function and terminal sets used. Some of these functions, such as the division, the exponential and the logarithm, have been implemented with a protection mechanism to avoid incomputable results (e. g. division by zero returns zero and the logarithm returns the logarithm of the absolute value of its argument, or 0 if the argument is 0).

3.2 Classification

The classification works as follows. Let $X = \{x_0, \ldots, x_{\mathrm{N}}\}$ be the vector comprising the data of the company undergoing classification. Let $f(X)$ be the function defined by an individual GP tree structure. The value y returned by $f(X)$ depends on the input vector X.

$$y = f(x_0, x_1, \ldots, x_{\mathrm{N}}) \tag{1}$$

We can apply X as the input to the GP tree and calculate the output y. Once the numerical value of y has been calculated, it will give us the classification result according to:

$$y > 0,\ X \in B \tag{2}$$

$$y \leq 0,\ X \in \overline{B} \tag{3}$$

where B represents the class to which bankrupt companies belong and $\overline{B}$ represents the class to which healthy companies belong.

That is, if the evaluation of the GP tree results in a numerical value greater than 0 the company is classified as heading for bankruptcy, while if the value is less or equal to 0 the company is classified as healthy.

3.3 Fitness Evaluation

As mentioned previously, the database we are using is very unbalanced in the sense that only 5-6% of the companies included will go bankrupt. This is something to take into account while designing the fitness function. Otherwise the evolution may converge to structures that classify all companies as healthy

(i.e. they do not classify at all) and still get a 95% hits rate. According to [10] there are 3 ways to address this problem:

- Undersampling the over-sized class
- Oversampling the small class
- Modifying the cost associated to misclassifying the positive and the negative class to compensate for the imbalanced ratio of the two classes. For example, if the imbalance ratio is 1:10 in favour of the negative class, the penalty of misclassifying a positive example should be 10 times greater.

We have used the cost-modifying approach, not only because it was the one recommended by the authors of [10] but mainly because the oversampling and undersampling approaches did not yield good results.

Therefore, the fitness function is:

$$Fitness = \sum_{i=1}^{n} u_i \tag{4}$$

where

$$u = \begin{cases} 0 & : \text{ incorrect classification} \\ 1 & : \text{ bankrupt company classified correctly} \\ \frac{n_{b=0}}{n_{b=1}} & : \text{ healthy company classified correctly} \end{cases} \tag{5}$$

$n_{b=0}$ is the number of bankrupt companies in the training set and $n_{b=1}$ is the number of healthy companies in the training set.

3.4 GP Algorithm

The GP implementation used is based on the JEO (Java Evolving Objects) library [3] developed within the project DREAM (Distributed Resource Evolutionary Algorithm Machine) [2]. The project's aim was to develop a complete distributed peer-to-peer environment for running evolutionary optimization applications over a set of heterogeneous distributed computers.

JEO is a software package in Java integrated in DREAM. In the context of GP, JEO includes a tree-shaped genome structure and several operators, therefore the user only needs to implement the methods that are problem dependent, i. e. fitness evaluation and construction of the function and terminal sets.

As a method of bloat control we have included a new crossover operator, bloat-control-crossover, that occurs with a probability of 0.45. This crossover operator implements a bloat control approach described in [9] and inspired in the "prune and plant" strategy used in agriculture. It is used mainly for fruit trees and it consist of pruning some branches of trees and planting them in order to grow new trees. The idea is that the worst tree in a population will be substituted by branches "pruned" from one of the best trees and "planted" in its place. This way the offspring trees will be of smaller size than the ancestors, effectively reducing bloat.

Table 2 shows the main parameters used during evolution.

Table 2. GP parameters

Initialization method	Ramped half and half
Replacement operator	Generational with elitism (0.2%)
Selection operator	Tournament selection
Tournament group size	10
Cloning rate	0.05
Crossover operator	Bias tree crossover
Internal node selection rate	0.9
Crossover rate	0.5
Crossover for "bloat" control rate	0.45
Tree maximum initial depth	7
Tree maximum depth	18
Population size	500
Number of runs	20
Termination criterion	50 generations

4 Results

The results obtained for each year are shown in the following tables. The first row of the tables shows the best result obtained and the second row shows the averaged results over 20 runs. Each table shows the results obtained in training, in testing and overall (i.e. training+testing). The first column shows the percentage of hits scored (i.e. the number of correct predictions), the second column presents the percentage of true positives (TP) (i.e. the number of companies heading for bankruptcy classified correctly) and the third column presents the percentage of true negatives (TN) (i.e. the number of healthy companies classified as such).

In the problem of bankruptcy prediction the results are very linked to the database in use so it is important to present results obtained with an alternative classification method for comparison purposes.

This section includes an alternative set of results obtained using a support vector machine (SVM). In order to generate these results we have used LIBSVM [6], an integrated software for support vector classification. We have chosen this particular software because it supports weighted SVM for unbalanced data. The SVM has been run 20 times using the same random training and testing sets as in the GP case.

4.1 Prediction of Bankruptcy in 2003 Using Data from Year 1999

Table 3 shows that the results obtained with GP in the training are very good. However, the average results in the testing show an average percentage of true positives smaller than 50%. This is due to GP converging to structures that achieve a good global result but at the expense of a very low percentage of true positives (i.e. the structure is classifying all companies as healthy). This explains why the average percentage of hits is greater than the one obtained by the best GP tree. Nevertheless, it can be seen that the best GP achieves

Table 3. GP results using data from year 1999

	Training			Testing			Overall		
	% hits	% TP	% TN	% hits	% TP	% TN	% hits	% TP	% TN
Best GP results	88.12	90.00	88.00	74.92	76.47	74.83	79.44	81.48	79.32
Avg. GP results	89.69	98.50	89.10	80.94	40.59	83.31	83.94	62.04	85.28

very satisfactory and balanced percentages of hits in the classification of both bankrupt and healthy companies.

The best GP individual can be expressed as follows:

$$
\begin{aligned}
y &= x_{29}\cos(40.97 - x_{28} + \exp(t_2))t_3 \\
t_1 &= \exp(x_9 x_{29}) \\
t_2 &= \cos(t_7 x_{29} + t_6(x_{29} + x_{20} + 40.97 - x_{28} + t_1)) \\
t_3 &= t_4 + x_{29}(\cos(2x_{23}) + \cos(x_{29}(40.97 - x_{28}) + t_1) + 81.94 + t_1 - x_{28} + t_5 \\
t_4 &= \exp(\exp(\cos(\cos(\exp(x_7) + x_{20}))\cos(40.97 - x_{28} + x_{23} + x_{20}))) \\
t_5 &= x_{20}\cos(t_1 + x_{20}) + \cos(40.97 - x_{28} + x_{26}) + \cos(x_{29}(t_6 + t_7) + x_{15}) \\
t_6 &= \cos(\exp(\exp x_7) + x_{20}) \\
t_7 &= \cos(\exp(\exp x_7) + x_{23})
\end{aligned}
\tag{6}
$$

When compared with the results obtained with SVM (see table 4), it can be seen that the GP results are superior.

Table 4. SVM results using data from year 1999

	Training			Testing			Overall		
	% hits	% TP	% TN	% hits	% TP	% TN	% hits	% TP	% TN
Best SVM results	64.37	90.00	62.67	71.01	94.12	69.66	68.74	92.59	62.27
Avg. SVM results	70.44	96.50	68.70	67.95	88.53	66.74	68.80	91.48	67.41

The percentage of TP is slightly better for the SVM, but the overall performance of the GP is better (79.44% versus 68.74% of hits) and more balanced. To further check if the difference in results between GP and SVM is real we have performed a Mann-Whitney U test to the testing results. The test has concluded that the results are statistically different.

Table 5. GP results using data from year 2000

	Training			Testing			Overall		
	% hits	% TP	% TN	% hits	% TP	% TN	% hits	% TP	% TN
Best GP results	77.50	100.00	76.00	73.04	80.00	72.57	74.53	86.67	73.72
Avg. GP results	89.81	99.50	89.17	79.94	43.75	82.36	83.24	62.33	84.63

4.2 Prediction of Bankruptcy in 2003 Using Data from Year 2000

The results obtained with GP using data from year 2000 are shown in table 5. The average results show that, due to the unbalanced data, there are some GP structures that do not achieve good percentages of true positives. The percentages of hits obtained with the best GP result are very satisfactory.

The best GP individual consists of 12 nested conditional clauses:

$$
\begin{aligned}
y &= \text{ if } x_{28} \leq x_{11} \text{ then } (f_0 - f_1)/x_{29}^2 \text{ else } f_2 \\
f_0 &= \text{ if } x_{26} \leq (f_4 - f_3) \text{ then } x_{29} \text{ else } x_{17} \\
f_1 &= \text{ if } x_{19} \leq x_4 \text{ then } x_{26} \text{ else } f_7 \\
f_2 &= \text{ if } f_8 \leq (x_{19} - x_{12} - x_3) \text{ then } x_{29} \text{ else } x_{17} \\
f_3 &= \text{ if } x_{19} \leq x_4 \text{ then } x_{26} \text{ else } f_5 \\
f_4 &= \text{ if } x_{26} \leq (f_6 - f_3) \text{ then } x_{29} \text{ else } x_{17} \\
f_5 &= \text{ if } x_{12} \leq x_{10} \text{ then } x_{28} \text{ else } x_5 \\
f_6 &= \text{ if } x_{26} \leq -4x_{12} \text{ then } x_{29} \text{ else } x_{17} \\
f_7 &= \text{ if } x_{12} \leq x_{27} \text{ then } x_{28} \text{ else } x_5 \\
f_8 &= \text{ if } x_{17} \leq x_{11} \text{ then } f_5/x_{29}^2 \text{ else } f_9 \\
f_9 &= \text{ if } f_5 \leq f_{10} \text{ then } x_{29} \text{ else } x_{17} \\
f_{10} &= \text{ if } x_{26} \leq (x_{18} - x_3 - 3x_{12}) \text{ then } x_{29} \text{ else } x_{17}
\end{aligned}
\tag{7}
$$

Again the overall performance of the GP is better (74.53% versus 69.10% of hits) and more balanced (see table 6). The Mann-Whitney U test has also confirm that the results obtained with GP and SVM are statistically different.

Table 6. SVM results using data from year 2000

	Training			Testing			Overall		
	% hits	% TP	% TN	% hits	% TP	% TN	% hits	% TP	% TN
Best SVM results	69.37	90.00	68.00	68.97	95.00	67.22	69.10	93.33	67.48
Avg. SVM results	69.28	98.50	67.33	67.29	87.25	65.95	67.95	91.00	66.41

5 Conclusions

The main problems we had to handle in this work was the imbalance between the number of companies heading for bankruptcy (around 5-6%) and the number of healthy companies, and the amount of missing data (around 16% of the companies have one or more missing data) in the database we have used for the analysis.

The approaches we have used to solve them have been the normalization of the data and the use of a fitness function that suited the unbalance problem.

The results obtained are very satisfactory. The best GP structure has achieved a percentage of hits of around 75% in the testing set. When compared with the numerical results obtained with SVM they are clearly better. In addition GP provides us with a nonlinear function in a tree shape, which is easier to analyze and draw conclusions from than the SVM black box structure.

In the future we plan to combine data from several years to carry out the prediction. This raises the problem of how to handle the data. We plan to use serial processing of data. Instead of presenting the data to the system as a vector (i.e. simultaneously) we will adopt an alternative approach in which the data from various years is presented to the classifier in series.

Acknowledgments

Thanks to Isabel Román Martínez, José M. de la Torre Martínez y Elena Gómez Miranda, from the Department of Finance and Accounting of the Universidad de Granada, for the financial database. This work was supported by the Plan Nacional de I+D of the Spanish Ministerio de Educación y Ciencia (NADEWeb project - TIC2003-09481-C04).

References

1. Altman, E. I.: Financial ratios, discriminant analysis and the prediction of corporate bankruptcy. J. of Finance **23 (4)** (1968) 589–609
2. Arenas, M. G., Collet, P.,Eiben, A. E., Jelasity, M., Merelo, J. J., Paechter, B., Preuß, M., Schoenauer, M.: "A framework for distributed evolutionary algorithms". Proc. of PPSN'02, in LNCS (Springer-Verlag) **2439** (2002) 665–675
3. Arenas, M. G., Dolin, B., Merelo, J. J., Castillo, P. A., Férnandez de Viana, I., Schoenauer, M.: "JEO: Java Evolving Objects". Proc. of the Genetic and Evolutionary Computation Conf., GECCO'02. New York, USA (2002) 991–994
4. Brabazon, A., Keenan, P. B.: A hybrid genetic model for the prediction of corporate failure. Computational Management Science **1** (2004) 293–310
5. Brabazon, A., O'Neill, M.: Biologically inspired algorithms for finantial modelling. Springer-Verlag, Berlin, 2006
6. Chang, C.-C., Lin, C.-J.: LIBSVM: a library for support vector machines. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm` (2001)
7. Dimitras, A. I., Zanakis, S. H., Zopounidis, C.: A survey of business failures with an emphasis on predictions, methods and industrial applications. Eur. J. Oper. Res. **90** (1996) 487–513

8. Eggermont, J., Eiben, A. E., van Hemert, J. I.: "A comparison of genetic programming variants for data classification". Proc. of IDAC'99, in LNCS (Springer-Verlag) **1642** (1999) 281–290
9. Fernández de Vega, F., Rubio del Solar, M., Fernández Martínez, A.: "Plantación de árboles: Una nueva propuesta para reducir esfuerzo en programación genética". Actas del IV Congreso Español sobre Metaheurísticas, Algoritmos Evolutivos y Bioinspirados, MAEB'05. Granada, Spain (2005) 57–62.
10. Japkowicz, N., Stephen, S.: The class imbalance problem: a systematic study. Intelligent Data Analysis **6:5** (2002) 429-449
11. Kim, M. J., Han, I.: The discovery of experts' decision rules from qualitative bankruptcy data using genetic algorithms. Expert Syst. Appl. **25** (2003) 637–646
12. Kishore, J. K., Patnaik, L. M., Mani, V., Agrawal, V. K.: Genetic programming based pattern classification with feature space partitioning. Inf. Sci. **131** (2001) 65–86
13. Lensberg, T., Eilifsen, A., McKee, T. E.: Bankruptcy theory development and classification via genetic programming. Eur. J. Oper. Res. **169** (2006) 677-697
14. Leshno, M., Spector, Y.: Neural network predction analysis: The bankruptcy case. Neurocomputing **10** (1996) 125–147
15. McKee, T. E., Lensberg, T.: Genetic programming and rough sets: A hybrid approach to bankruptcy classification. Eur. J. Oper. Res. **138** (2002) 436–451
16. Ohlson, J.: Financial ratios and the probabilistic prediction of bankruptcy. J. of Accounting Research **18 (1)** (1980) 109–131
17. Salcedo-Sanz, S., Fernández-Villacañas, J. L., Segovia-Vargas, M. J. Bousoño-Calzón, C.: Genetic programming for the prediction of insolvency in non-life insurance companies. Computers & Operations Research **32** (2005) 749–765
18. Shin, K. S., Lee, Y. L.: A genetic algorithm application in bankruptcy prediction modeling. Expert Syst. Appl. **23** (2002) 321–328
19. Tsakonas, A., Dounias, G., Doumpos, M., Zopounidis, C.: Bankruptcy prediction with neural logic networks by means of grammar-guided genetic programming. Expert Syst. Appl. **30** (2006) 449-461
20. Varetto, F.: Genetic algorithm applications in the field of insolvency risk. Journal of Banking and Finance **22** (1998) 1421–1439
21. Vieira, A. S., Ribeiro, B., Mukkamala, S., Neves, J. C., Sung, A. H.: "On the performance of learning machines for bankruptcy detection". Proc. of the IEEE Conf. on Computational Cybernetics. Vienna, Austria (2004) 323–327

Multi-objective Optimization Technique Based on Co-evolutionary Interactions in Multi-agent System

Rafał Dreżewski and Leszek Siwik

Department of Computer Science
AGH University of Science and Technology, Kraków, Poland
{drezew,siwik}@agh.edu.pl

Abstract. Co-evolutionary techniques for evolutionary algorithms help overcoming limited adaptive capabilities of evolutionary algorithms, and maintaining population diversity. In this paper the idea and formal model of agent-based realization of predator-prey co-evolutionary algorithm is presented. The effect of using such approach is not only the location of Pareto frontier but also maintaining of useful population diversity. The presented system is compared to classical multi-objective evolutionary algorithms with the use of Kursawe test problem and the problem of effective portfolio building.

1 Introduction

Co-evolutionary techniques for *evolutionary algorithms (EAs)* are applicable in the case of problems for which the fitness function formulation is difficult or impossible, there is need for improving adaptive capabilities of EA or maintaining useful population diversity and introducing speciation into EAs—loss of population diversity is one the main problems in some applications of EAs (for example multi-modal optimization, multi-objective optimization, dynamic problems, etc.)

In the case of multi-objective optimization problems loss of population diversity may cause that the population locates in the areas far from Pareto frontier or that individuals are located only in selected areas of Pareto frontier. In the case of multi-objective problems with many local Pareto frontiers (defined by Deb in [2]) the loss of population diversity may result in locating only local Pareto frontier instead of a global one.

One of the first attempts to apply competitive co-evolutionary algorithm to multi-objective problems was predator-prey evolutionary strategy (PPES) [6]. This algorithm was then modified by Deb [2] in order to introduce some mechanisms of maintaining population diversity and evenly distributing individuals over the Pareto frontier, but this is still an open issue and the subject of ongoing research.

Evolutionary multi-agent systems (EMAS) are multi-agent systems, in which the population of agents evolve (agents can die, reproduce and compete for limited resources). The model of *co-evolutionary multi-agent system (CoEMAS)* [3] introduces additionally the notions of species, sexes, and interactions between them. CoEMAS allows modeling and simulation of different co-evolutionary interactions, which can serve as the basis for constructing the techniques of maintaining population diversity and improving adaptive capabilities of such systems. CoEMAS systems with sexual selection and host-parasite

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 179–188, 2007.
© Springer-Verlag Berlin Heidelberg 2007

mechanisms have already been applied with good results to multi-objective optimization problems ([4,5]). In the following sections the introduction to multi-objective optimization problems is presented. Next, the co-evolutionary multi-agent system with predator-prey mechanism is formally described. The system is applied to one standard multi-objective optimization test problem and to problem of effective portfolio building. Results from the experiments with the CoEMAS system are then compared to other classical evolutionary techniques' results.

2 Multi-objective Optimization

Multi-criteria Decision Making (MCDM) is the most natural way of making decision for human beings. *Multi-criteria* means that during the decision process a lot of factors and objectives (often contradictory) are taken into consideration. Human being is equipped with natural gifts for multi-criteria decision making, however such abilities are not sufficient in more complex technical, business or scientific decisions. In such cases decision maker has to be equipped with efficient information systems able to support his decision making process.

MCDM process is based most frequently on *Multi-objective Optimization* formulated formally only in 19th century, but actual progress in solving *Multi-objective Optimization Problems (MOOP)* ensued after formulating by Vilfredo Pareto his optimality theory in 1906. Following [2]—*Multi-objective Optimization Problem* in its general form can be formulated as follows:

$$MOOP \equiv \begin{cases} Minimize/Maximize \;\; f_m(\bar{x}), & m = 1,2\ldots,M \\ Subject\ to \;\; g_j(\bar{x}) \geq 0, & j = 1,2\ldots,J \\ \qquad\qquad h_k(\bar{x}) = 0, & k = 1,2\ldots,K \\ \qquad\qquad x_i^{(L)} \leq x_i \leq x_i^{(U)}, & i = 1,2\ldots,N \end{cases}$$

The set of constraints—both constraint functions (equalities $h_k(\bar{x})$) and inequalities $g_k(\bar{x})$) and decision variable bounds (lower bounds $x_i^{(L)}$ and upper bounds $x_i^{(U)}$)—define all possible (feasible) decision alternatives ($\mathcal{D}$).

The crucial concept of Pareto optimality is so called dominance relation that can be formulated as follows: to avoid problems with converting minimization to maximization problems (and vice versa of course) additional operator $\lhd$ can be introduced. Then, notation $\bar{x}_1 \lhd \bar{x}_2$ indicates that solution $\bar{x}_1$ is simply better than solution $\bar{x}_2$ for particular objective. It is said that solution $\bar{x}_A$ dominates solution $\bar{x}_B$ ($\bar{x}_A \prec \bar{x}_B$) then and only then if:

$$\bar{x}_A \prec \bar{x}_B \Leftrightarrow \begin{cases} f_j(\bar{x}_A) \not\rhd f_j(\bar{x}_B) & for \quad j = 1,2\ldots,M \\ \exists i \in \{1,2,\ldots,M\} \;\; : \;\; \bar{x}_A \lhd \bar{x}_B \end{cases}$$

A solution in the Pareto sense of the multi-objective optimization problem means determination of all non-dominated alternatives from the set $\mathcal{D}$.

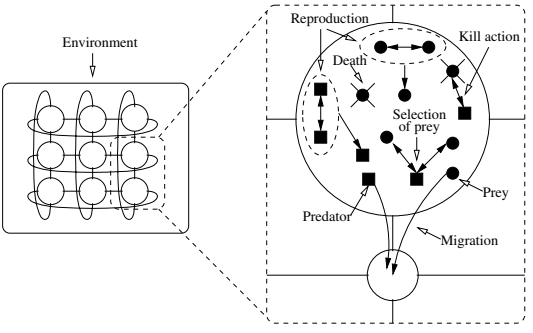


Fig. 1. CoEMAS with predator-prey mechanism

3 Co-evolutionary Multi-agent System with Predator-Prey Mechanism for Multi-objective Optimization

The system presented in this paper is based on the CoEMAS model, which is the general model of co-evolution in multi-agent system. In order to maintain population diversity predator-prey co-evolutionary mechanism is used (see fig. 1). Prey represent solutions of the multi-objective problem. The main goal of predators is to eliminate "weak" (ie. dominated) prey.

The **co-evolutionary multi-agent system with predator-prey mechanism** is described as 4-tuple: $CoEMAS = \langle E, S, \Gamma, \Omega \rangle$ where S is the set of species ($s \in S$) that co-evolve in $CoEMAS$, Γ is the set of resource types that exist in the system, the amount of type γ resource will be denoted by r^{γ}, Ω is the set of information types that exist in the system, the information of type ω will be denoted by i^{ω}.

$E = \left\langle T^E, \Gamma^E, \Omega^E \right\rangle$ is the **environment** of the $CoEMAS$, where T^E is the topography of the environment E (directed graph with the cost function defined), Γ^E is the set of resource types that exist in the environment—in our case $\Gamma^E = \Gamma$, Ω^E is the set of information types that exist in the environment—in the described system $\Omega^E = \Omega$.

There are two **information types** ($\Omega = \{\omega_1, \omega_2\}$) and one resource type ($\Gamma = \{\gamma\}$) in *CoEMAS*. Informations of type ω_1 contain nodes to which agent can migrate, when it is located in particular node of the graph. Informations of type ω_2 contain agents-prey which are located in the particular node in time t. There is one **resource type** ($\Gamma = \{\gamma\}$) in *CoEMAS*, and there is closed circulation of resource within the system.

The **set of species** is given by: $S = \{prey, pred\}$. The prey species is defined as follows: $prey = \langle A^{prey}, SX^{prey} = \{sx\}, Z^{prey}, C^{prey} \rangle$, where A^{prey} is the set of agents that belong to the *prey* species, SX^{prey} is the set of sexes which exist within the *prey* species, Z^{prey} is the set of actions that agents of species *prey* can perform, and C^{prey} is the set of relations of species *prey* with other species that exist in the $CoEMAS$. There is only one sex sx ($sx \equiv sx^{prey}$) within the *prey* species, which is defined as follows: $sx = \langle A^{sx} = A^{prey}, Z^{sx} = Z^{prey}, C^{sx} = \emptyset \rangle$.

The **set of actions** $Z^{prey} = \{die, get, give, accept, seek, clone, rec, mut, migr\}$, where *die* is the action of death, which is performed when prey is out of resources, *get* action

gets some resource from another a^{prey} agent located in the same node (this agent must be dominated by the agent that performs *get* action or is too close to him in the criteria space—*seek* action allows to find such agents), *give* actions gives some resource to another agent (which performs *get* action), *accept* action accepts partner for reproduction (partner is accepted when the amount of resource possessed by the prey agent is above the given level), *seek* action also allows the prey agent to find partner for reproduction when the amount of its resource is above the given level, *clone* is the action of cloning prey (new agent with the same genotype as parent's one is created), *rec* is the recombination operator (intermediate recombination is used [1]), *mut* is the mutation operator (mutation with self-adaptation is used [1]), *migr* action allows prey to migrate between the nodes of the graph (migrating agent loses some resource).

The **set of relations of** *prey* **species** with other species that exist within the system is defined as follows: $C^{prey} = \left\{ \xrightarrow{prey,get-} = \{\langle prey, prey\rangle\}, \xrightarrow{pred,give+} = \{\langle prey, pred\rangle\} \right\}$. The first relation models intra species competition for limited resources (prey can decrease ("-") the fitness of another prey with the use of *get* action). The second one models predator-prey interactions: prey gives all the resource it owes to predator (which fitness is increased: "+") and then dies.

The **predator species** (*pred*) is defined analogically as *prey* species with the following differences. The set of actions $Z^{pred} = \{seek, get, migr\}$, where *seek* action seeks for the "worst" (according to the criteria associated with the given predator) prey located in the same node, *get* action gets all resource from chosen prey, *migr* action is analogical as in the case of prey species. The **set of relations of** *pred* **species** with other species is limited to one relation, which models predator-prey interactions: $C^{pred} = \left\{ \xrightarrow{prey,get-} = \{\langle pred, prey\rangle\} \right\}$.

Agent a **of species** *prey* is given by: $a = \langle gn^a, Z^a = Z^{prey}, \Gamma^a = \Gamma, \Omega^a = \Omega, PR^a \rangle$. Genotype gn^a is consisted of two vectors (chromosomes): x of real-coded decision parameters' values and σ of standard deviations' values, which are used during mutation. $Z^a = Z^{prey}$ is the set of actions which agent a can perform. Γ^a is the set of resource types, and Ω is the set of information types, which agent can possess.

The set of profiles PR^a includes resource profile (pr_1, which goal is to maintain the amount of resource above the minimal level), reproduction profile (pr_2, which goal is agent's reproduction), interaction profile (pr_3, which goal is to interact with agents from the same and another species), and migration profile (pr_4, which goal is to migrate to another node). Each time step agent tries to realize goals of the profiles taking into account their priorities: $pr_1 \trianglelefteq pr_2 \trianglelefteq pr_3 \trianglelefteq pr_4$ (pr_1 has the highest priority). In order to realize goal of the given profile agent uses actions which can be realized within the given profile. For example within pr_1 profile all actions connected with type γ resource (*die*, *seek*, *get*) can be used in order to realize the goal of this profile. This profile uses informations of type ω_2.

Agent a **of species** *pred* is defined analogically to *prey* agent. The main differences are genotype and the set of profiles. Genotype of agent a is consisted of the information about the criterion associated with this agent. The set of profiles PR^a includes only resource profile (pr_1, which goal is to "kill" prey and collect their resources), and migration profile (pr_2, which goal is to migrate within the environment).

4 Test Problems

The experimental and comparative studies presented in this paper are based on well known *Kursawe* multi-objective test problem (the formal definition may be found in [7]) and the problem of effective portfolio building.

The Pareto set and Pareto frontier for the *Kursawe* problem are presented in fig. 2. In this case optimization algorithm has to deal with disconnected two-dimensional Pareto frontier and disconnected three dimensional Pareto set. Additionally, a specific definition of f_1 and f_2 functions causes that even very small changes in the space of decision variables can cause big differences in the space of objectives. All of these causes that *Kursawe* problem is quite difficult for solving in general—and for solving using evolutionary techniques in particular.

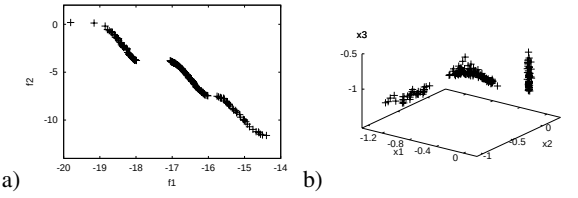


Fig. 2. *Kursawe test problem*: a) Pareto frontier and b) Pareto set

Proposed co-evolutionary agent-based approach has also been preliminarily assessed using the problem of effective portfolio building. Below, there are presented consecutive steps (based on the Sharp model) during computing the expectation of the risk level and generally speaking income expectation related to the wallet of p shares: 1. Computing of arithmetic means on the basis of rate of returns; 2. Computing the value of α coefficient: $\alpha_i = \overline{R_i} - \beta_i \overline{R_m}$, where R_i is the rate of return of i-th share, and R_m is the rate of return of market index; 3. Computing the value of β coefficient: $\beta_i = \frac{\sum_{t=1}^{n}(R_{it}-\overline{R_i})(R_{mt}-\overline{R_m})}{\sum_{t=1}^{n}(R_{mt}-\overline{R_m})^2}$, where n is the number of rate of return, R_{it} is the rate of return in the period t, R_{mt} is the rate of return related to market index in period t; 4. Computing the share expectation: $R_i = \alpha_i - \beta_i R_m + e_i$, where e_i is the random component of the equation; 5. Computing the variance of random index of the $i-th$ share: ${s_{e_i}}^2 = \frac{\sum_{t=1}^{n}(R_{it}-\alpha_i-\beta_i R_m)^2}{n-1}$; 6. Computing the variance of market index: ${s_m}^2 = \frac{\sum_{t=1}^{n}(R_{mt}-\overline{R_m})^2}{n-1}$; 7. Computing the risk level of the investing wallet: $risk = \beta_p^2 {s_m}^2 + {s_{e_p}}^2$, where $\beta_p = \sum_{i=1}^{p}(\omega_i \beta_i)$, p is the number of shares in the wallet, ω_i is the percentage participation of i-th share in the wallet, ${s_{e_p}}^2 = \sum_{i=1}^{p}(\omega_i^2 {s_{e_i}}^2)$ is the variance of the wallet; 8. Computing the investing wallet expectation: $R_p = \sum_{i=1}^{p}(\omega_i R_i)$. The goal of optimization is to maximize the investing wallet expectation along with minimizing the risk level. Model Pareto frontiers related to two cases taken into consideration in the course of this paper are presented in fig 3.

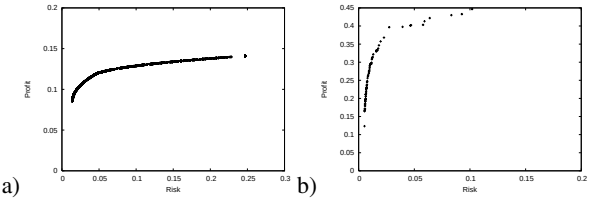


Fig. 3. *Building of effective portfolio*: the model Pareto frontier for a) 3 and b) 17 stocks set

5 Results of Experiments

As it was mentioned in sec. 4 proposed CoEMAS system with predator-prey mechanism has been evaluated using inter-alia Kursawe test problem. To give a kind of reference point, results obtained by CoEMAS are compared with results obtained by "classical" (i.e. non agent-based) predator-prey evolutionary strategy (PPES) [6] and another classical evolutionary algorithm for multi-objective optimization: niched pareto genetic algorithm (NPGA) [2]. In fig. 4 approximations of Pareto frontier obtained by all three algorithms are presented. As one may notice initially, i.e. after 1, 10 and partially after 20 (see fig. 4a, 4b and 4c) steps, Pareto frontiers obtained by all three algorithms are quite similar if the number of found non-dominated individuals, their distance to the model Pareto frontier and their dispersing over the whole Pareto frontier are considered. Afterwards yet, definitely higher quality of CoEMAS-based Pareto frontier approximation is more and more distinct. The NPGA-based Pareto frontier almost completely disappears after about 30 steps, and although PPES-based Pareto frontier is better and better this improving process is quite slow and not so clear as in the case of CoEMAS-based solution.

Because solutions presented in fig. 4 partially overlap, in fig. 5 there are presented separately Pareto frontiers obtained by analyzed algorithms after 2000, 4000 and 6000

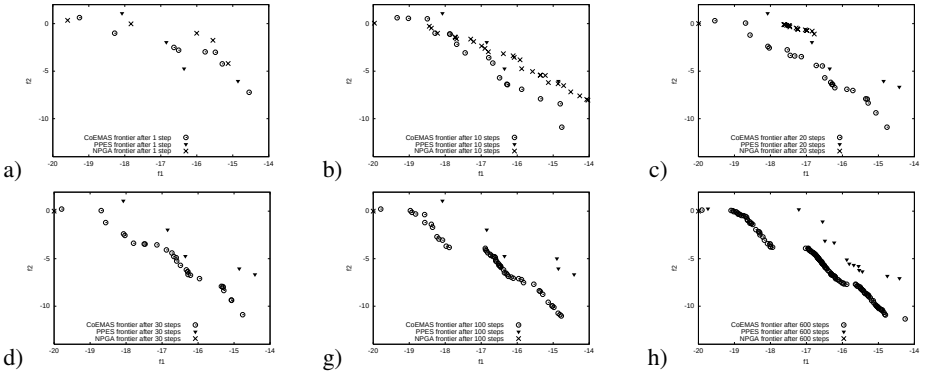


Fig. 4. Kursawe problem Pareto frontier approximations obtained by CoEMAS, PPES and NPGA after a) 1, b) 10, c) 20, d) 30, e) 100 and f) 600 steps

Table 1. The values of the *HV* and *HVR* metrics for compared systems (Kursawe problem)

	HV / HVR		
Step	**CoEMAS**	**PPES**	**NPGA**
1	541.21 / 0.874	530.76 / 0.857	489.34 / 0.790
10	588.38 / 0.950	530.76 / 0.867	563.55 / 0.910
20	594.09 / 0.959	531.41 / 0.858	401.79 / 0.648
30	601.66 / 0.971	531.41 / 0.858	378.78 / 0.611
40	602.55 / 0.973	531.41 / 0.858	378.73 / 0.611
50	594.09 / 0.959	531.41 / 0.858	378.77 / 0.611
100	603.04 / 0.974	531.42 / 0.858	378.80 / 0.6117
600	603.79 / 0.975	577.44 / 0.932	378.80 / 0.611
200	611.43 / 0.987	609.47 / 0.984	378.80 / 0.611
4000	611.44 / 0.987	555.53 / 0.897	378.80 / 0.611
6000	613.10 / 0.990	547.73 / 0.884	378.80 / 0.611

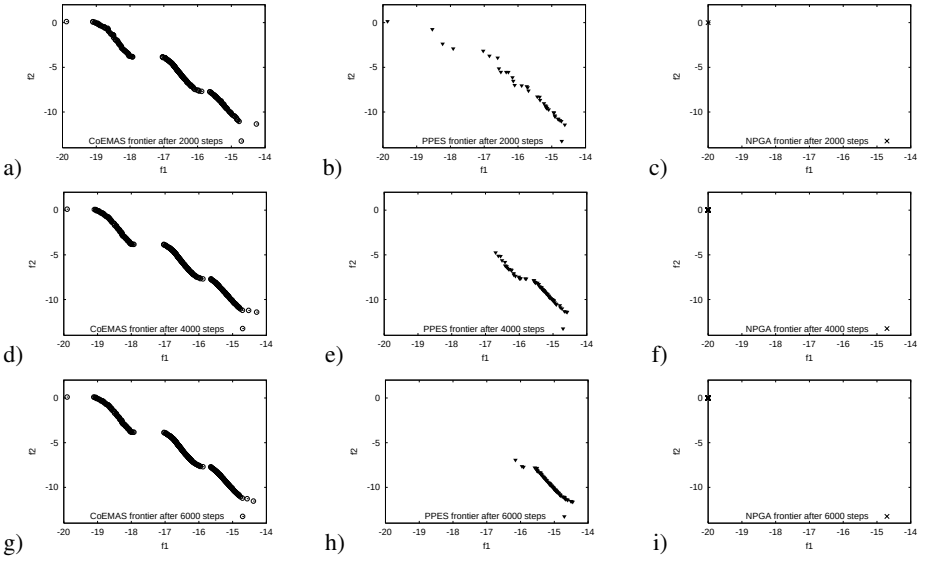


Fig. 5. Kursawe problem Pareto frontier approximations after 2000 (a), (b), (c), 4000 (d), (e),(f) and 6000 (g), (h), (i) steps obtained by CoEMAS, PPES, and NPGA

time steps. There is no doubt that—what can be especially seen in fig. 5a, d and g—CoEMAS is definitely the best alternative since it is able to obtain Pareto frontier that is located very close to the model solution, that is very well dispersed and what is also very important—it is more numerous than PPES and NPGA-based solutions.

It is of course quite difficult to compare algorithms only on the basis of qualitative results, so in Table 1 there are presented values of HV and HVR metrics ([2]) obtained during the experiments with Kursawe problem. The results presented in this

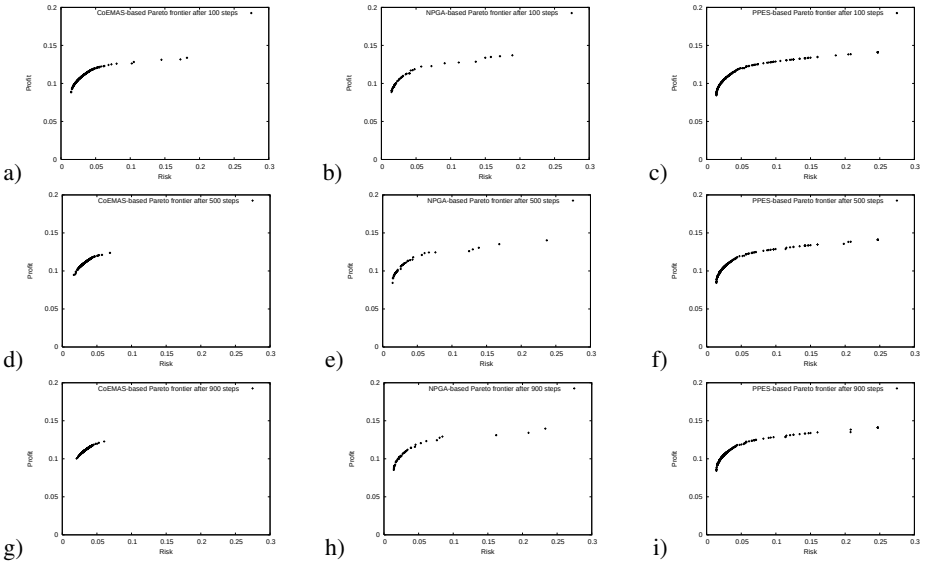


Fig. 6. Pareto frontier approximations after 100 (a), (b), (c), 500 (d), (e),(f), and 900 (g), (h), (i) steps obtained by CoEMAS, PPES, and NPGA for building effective portfolio consisting of 3 stocks

table confirm that in the case of Kursawe problem CoEMAS is much better alternative than "classical" PPES or NPGA algorithms.

In the case of optimizing investing portfolio each individual in the prey population is represented as a p-dimensional vector. Each dimension represents the percentage participation of i-th ($i \in 1 \ldots p$) share in the whole portfolio. Because of the space limitation in this paper only a kind of summary of two single experiments will be presented. During presented experiment quotations from 2003-01-01 until 2005-12-31 were taken into consideration. Simultaneously the portfolio consists of the following three (in experiment I) or seventeen (in experiment II) stocks quoted on the Warsaw Stock Exchange: in experiment I: RAFAKO, PONARFEH, PKOBP, in experiment II: KREDYTB, COMPLAND, BETACOM, GRAJEWO, KRUK, COMARCH, ATM, HANDLOWY, BZWBK, HYDROBUD, BORYSZEW, ARKSTEEL, BRE, KGHM, GANT, PROKOM, BPHPBK. As the market index WIG20 has been taken into consideration. In fig. 6 there are presented Pareto frontiers obtained using CoEMAS, NPGA and PPES algorithm after 100, 500 and 900 steps in experiment I. As one may notice in this case CoEMAS-based frontier is more numerous (especially initially) than NPGA-based and as numerous as PPES-based one. Unfortunately in this case diversity of population in CoEMAS approach is visibly worse than in the case of NPGA or PPES-based frontiers [1]. What is more, with time the tendency of CoEMAS-based solver for focusing solutions around small part of the whole Pareto frontier is more and more

[1] It is also confirmed by values of HV or HVR metrics, but because of space limitations these characteristics are omitted in this paper.

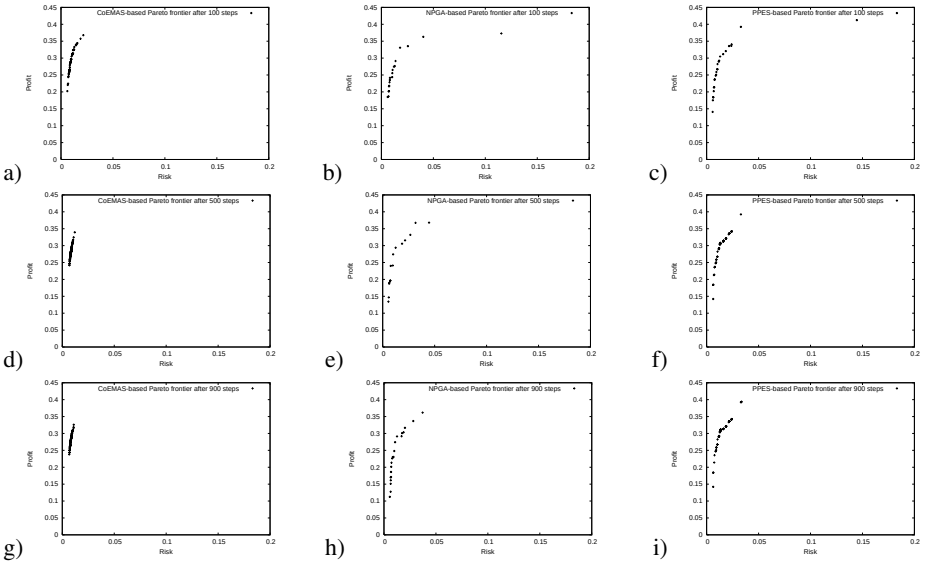


Fig. 7. Pareto frontier approximations after 100 (a), (b), (c), 500 (d), (e),(f), and 900 (g), (h), (i) steps obtained by CoEMAS, PPES, and NPGA for building effective portfolio consisting of 17 stocks

distinct. Similar situation can be also observed in fig. 7 presenting Pareto frontiers obtained by CoEMAS, NPGA and PPES—but this time portfolio that is being optimized consists of 17 shares. Also this time CoEMAS-based frontier is quite numerous and quite close to the model Pareto frontier but the tendency for focusing solutions around only selected part(s) of the whole frontier is very distinct [2].

6 Concluding Remarks

Co-evolutionary techniques for evolutionary algorithms are applicable in the case of problems for which it is difficult or impossible to formulate explicit fitness function, there is need for maintaining useful population diversity, forming species located in the basins of attraction of different local optima, or introducing open-ended evolution. Such techniques are also widely used in artificial life simulations. Although co-evolutionary algorithms has been recently the subject of intensive research their application to multimodal and multi-objective optimization is still the open problem and many questions remain unanswered.

In this paper the agent-based realization of predator-prey model within the more general framework of *co-evolutionary multi-agent system* has been presented. The system was run against Kursawe test problem and hard real-life multi-objective problem—

[2] It is also confirmed by values of appropriate metrics but as it was said those characteristics are omitted in this paper.

effective portfolio building—and then compared to two classical multi-objective evolutionary algorithms: PPES and NPGA. In the case of difficult Kursawe test problem CoEMAS with predator-prey mechanism properly located Pareto frontier, the useful population diversity was maintained and the individuals were evenly distributed over the whole frontier. In the case of this test problem the results obtained with the use of proposed system was clearly better than in the case of two other "classical" algorithms. It seems that the proposed predator-prey mechanism for evolutionary multi-agent systems may be very useful in the case of hard dynamic and multi-modal multi-objective problems (as defined by Deb [2]). In the case of effective portfolio building problem CoEMAS was able to form more numerous frontier, however negative tendency to lose population diversity during the experiment was observed. In this case PPES and NPGA were able to form better dispersed Pareto frontiers. The results of experiments show that still more research is needed on co-evolutionary mechanisms for maintaining population diversity used in CoEMAS, especially when we want to stably maintain diversity of solutions. Future work will include more detailed analysis of proposed co-evolutionary mechanisms, especially focused on problems of stable maintaining population diversity. Also the comparison of CoEMAS to other classical multi-objective evolutionary algorithms with the use of hard multi-modal multi-objective test problems, and the application of other co-evolutionary mechanisms like symbiosis (co-operative co-evolution) are included in future plans.

References

1. T. Bäck, editor. *Handbook of Evolutionary Computation*. IOP Publishing and Oxford University Press, 1997.
2. K. Deb. *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley & Sons, 2001.
3. R. Dreżewski. A model of co-evolution in multi-agent system. In V. Mařík, et al., editor, *Multi-Agent Systems and Applications III*, volume 2691 of *LNCS*, Berlin, Heidelberg, 2003. Springer-Verlag.
4. R. Dreżewski and L. Siwik. Co-evolutionary multi-agent system with sexual selection mechanism for multi-objective optimization. In *Proc. of the IEEE World Congress on Computational Intelligence (WCCI 2006)*. IEEE, 2006.
5. R. Dreżewski and L. Siwik. Multi-objective optimization using co-evolutionary multi-agent system with host-parasite mechanism. In V. N. Alexandrov, et al., editor, *Computational Science — ICCS 2006*, volume 3993 of *LNCS*, Berlin, Heidelberg, 2006. Springer-Verlag.
6. M. Laumanns, G. Rudolph, and H.-P. Schwefel. A spatial predator-prey approach to multi-objective optimization: A preliminary study. In A. E. Eiben, et al., editor, *Parallel Problem Solving from Nature — PPSN V*, volume 1498 of *LNCS*. Springer-Verlag, 1998.
7. D. A. Van Veldhuizen. *Multiobjective Evolutionary Algorithms: Classifications, Analyses and New Innovations*. PhD thesis, Graduate School of Engineering of the Air Force Institute of Technology Air University, 1999.

Quantum-Inspired Evolutionary Algorithms for Calibration of the VG Option Pricing Model

Kai Fan[1], Anthony Brabazon[1], Conall O'Sullivan[2], and Michael O'Neill[1]

[1] Natural Computing Research and Applications Group
University College Dublin, Ireland
kai.fan@ucd.ie, Anthony.Brabazon@ucd.ie, Michael.ONeill@ul.ie
[2] School of Business, University College Dublin, Ireland
conall.osullivan@ucd.ie

Abstract. Quantum effects are a natural phenomenon and just like evolution, or immune systems, can serve as an inspiration for the design of computing algorithms. This study illustrates how a quantum-inspired evolutionary algorithm can be constructed and examines the utility of the resulting algorithm on Option Pricing model calibration. The results from the algorithm are shown to be robust and comparable to those of other algorithms.

1 Introduction

The objective of this study is to illustrate the potential for using a quantum rather than a traditional encoding representation in an evolutionary algorithm, and also to assess the utility of the resulting algorithm for the purposes of calibrating an option pricing model. This purpose of this paper is to test the QIEA on a relatively simple option pricing model with the intention of testing the algorithm on more comprehensive option pricing models using more option data at a later stage.

In recent years there has been a substantial interest in the theory and design of quantum computers, and the design of programs which could run on such computers. One interesting strand of research has been the use of natural computing (for example GP) to generate quantum circuits or programs (algorithms) for quantum computers [1]. There has also been associated work in a reverse direction which draws inspiration from concepts in quantum mechanics in order to design novel natural computing algorithms. This is currently an area of active research interest. For example, quantum-inspired concepts have been applied to the domains of evolutionary algorithms [2,3,4,5,6], social computing [8], neurocomputing [9,10,11], and immuno-computing [12,13]. A claimed benefit of these algorithms is that because they use a quantum representation, they can maintain a good balance between exploration and exploitation. It is also suggested that they offer computational efficiencies as use of a quantum representation can allow the use of smaller population sizes than typical evolutionary algorithms.

Quantum-inspired algorithms offer interesting potential. As yet, due to their novelty, only a small number of recent papers have implemented a QEA, typically

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 189–198, 2007.
© Springer-Verlag Berlin Heidelberg 2007

reporting good results [5,6]. Consequently, we have a limited understanding of the performance of these algorithms and further testing is required in order to determine both their effectiveness and their efficiency. It is also noted that although a wide-variety of biologically-inspired algorithms have been applied for financial modelling [7], the QEA methodology has not yet been applied to the finance domain. This study addresses both of these gaps.

2 The Quantum-Inspired Genetic Algorithm

The best-known application of quantum-inspired concepts in evolutionary computing is the quantum-inspired genetic algorithm (QIGA) [2,5,6]. The (QIGA) is based on the concepts of a qubit (quantum bit) and the superposition of states. In essence, in QIGAs the traditional representations used in evolutionary algorithms (binary, numeric and symbolic) are extended to include a quantum representation. Under a quantum representation, the basic unit of information is no longer a bit which can assume two distinct states (0 or 1), but is a quantum system. Hence, a qubit (the smallest unit of information in a two-state quantum system) can assume either of the two ground states (0 or 1) or any superposition of the two ground states (the quantum superposition). A qubit can therefore be represented as

$$|q^i\rangle = \alpha|0\rangle + \beta|1\rangle \tag{1}$$

where $|0\rangle$ and $|\rangle$ are the ground states 0 and 1, and α & β are complex numbers that specify the probability amplitudes of the two ground states. The act of observing (or measuring) a qubit projects the quantum system onto one of the ground states. $|\alpha|^2$ is the probability that the qubit will be in state 0 when it is observed, and $|\beta|^2$ is the probability that it will be in state 1. Hence, a qubit encodes the *probability* that a specific ground state will be seen when an observation takes place, rather than encoding the ground states themselves. In order to ensure this probabilistic interpretation remains valid, the values for α and β are constrained such that $|\alpha|^2 + |\beta|^2 = 1$.

More generally, a quantum system of m qubits can represent a total of 2^m states simultaneously. In the language of evolutionary computation a system of m qubits can be referred to as a *quantum chromosome* and can be written as a matrix

$$\begin{bmatrix} \alpha_1 & \alpha_2 & \dots & \alpha_m \\ \beta_1 & \beta_2 & \dots & \beta_m \end{bmatrix} \tag{2}$$

A key point when considering quantum systems is that they can compactly convey information on a large number of possible system states. In classical bit strings, a string of length n can represent 2^n possible states. However, a quantum space of n qubits has 2^n *dimensions*. This means that even a short qubit can convey information on many possible system states. For example, a 3 bit quantum system can encode 8 (2^3) distinct binary strings, and an 8 bit quantum system can encode 256 distinct strings. Due to its probabilistic interpretation, a single qubit of length m can simultaneously represent *all* possible bit strings of length

2^m. This implies that it is possible to modify standard evolutionary algorithms to work with a single quantum individual, rather than having to use a population of solution encodings. The qubit representation of the system states can also help maintain diversity during the search process of an evolutionary algorithm, due to its capability to represent multiple system states simultaneously.

2.1 The Algorithm

There is no single QIGA, rather there are a family of possible algorithms which could be derived from the joint quantum-evolutionary metaphor. However, the following algorithm provides an example of a canonical QIGA

```
Set t=0
Initalise Q(t)

Create P(t) by undertaking an observation of Q(t)

Evaluate P(t) and select the best solution

Store the best solution in P(t) into B(t)

While (t < max t)
    t=t+1
    Create P*(t) by undertaking observations of Q(t-1)
    Evaluate P*(t)
    Update Q(t)
    Store the best solutions in B(t-1) and P(t) into B(t)
Endwhile
```

Initially, the population of quantum chromosomes is created

$$Q(t) = q_1(t), q_2(t), \ldots, q_n(t),$$

where n is the population size, and each member of the population consists of an individual qubit of length m. The α and β values for each qubit are set to $\frac{1}{\sqrt{2}}$ in order to ensure that the states 0 and 1 are equally likely for each qubit.If there is domain knowledge that some states are likely to lead to better results, this can be used to seed the initial quantum chromosome(s). Once a population of quantum chromosomes are created, these can be used to create a population of binary (or solution encoding) strings by performing an 'observation' on the quantum chromosomes. One way of performing the observation step is to draw a random number $rnd \in [0, 1]$. If $rnd > |\alpha_i(t)|^2$, the corresponding bit (j) in $p_i^j(t)$ is assigned state 1, otherwise it is assigned state 0. Due to the stochastic nature of the observation step, the QIGA could be implemented using a single quantum chromosome, where this chromosome is observed multiple times in order to generate the population $P(t) = p_1(t), p_2(t), \ldots, p_i(t)$. Alternatively, a small population of quantum chromosomes could be maintained, with each chromosome being observed a fixed number of times in order to generate $P(t)$. In the while loop, an update step is performed on the quantum chromosome(s). This update step could be performed in a variety number of ways, for example by using pseudo-genetic operators, or by using a suitable quantum gate [3]. However the step is undertaken, its essence is that the quantum chromosome is adjusted

in order to make the generation of the best solution found so far, more likely in the next iteration. As the optimal solution is approached by the QIGA system, the values of each element of the quantum chromosome tend towards either 0 or 1, corresponding to a high probability that the quantum chromosome will generate a specific solution vector (p_i) when observed.

Quantum Mutation. Quantum mutation is loosely inspired by the standard GA mutation operator. However, this is adapted so that the mutation step is guided by the best individual found to date, with the quantum chromosome being altered in order to make the generation of this solution more likely in future iterations of the algorithm [5,6].

$$Q_{pointer}(t) = a * B_{bestsolution}(t) + (1 - a) * (1 - B_{bestsolution}(t)) \tag{3}$$

$$Q(t + 1) = Q_{pointer}(t) + b * randnorm(0, 1) \tag{4}$$

where $B_{bestsolution}(t)$ is the best solution found by iteration t. $Q_{pointer}(t)$ is a temporary quantum chromosome which is used to guide the generation of $Q(t+1)$ towards the form of $B_{bestsolution}$. The term $randnorm(0, 1)$ is a random number drawn from a (0,1) normal distribution. The parameters a and b control the balance between exploration and exploitation, with a governing the importance attached to $B_{bestsolution}(t)$ and b governing the degree of variance generation, centred on $Q_{pointer}(t)$. Values of $a \in [0.1, 0.5]$ and $b \in [0.05, 0.15]$ are suggested by [5,6].

3 Option Pricing Model Calibration

An optimisation problem in financial modelling is considered to test the performance of the QIGA. The optimisation involves calibrating an option pricing model to observed market data. Calibration is a method of choosing model parameters so that the distance between a set of model option prices and market option prices is minimised, where distance is some metric such as the sum of squared errors or the sum of squared percentage errors. The parameters can be thought to resemble the market's view on current option prices and the underlying asset price. In calibration we do not explicitly take into account any historical data. All necessary information is contained in today's option prices which can be observed in the market. Practitioners frequently calibrate option pricing models so that the models provides a reasonable fit to current observed market option prices and they then use these models to price exotic derivatives or for hedging purposes. In this paper we calibrate a popular extension of the Black-Scholes [16] option pricing model known as the Variance Gamma (VG) model [17,18,19] to FTSE 100 index option data.

A European call option on an asset S_t with maturity date T and strike price K is defined as a contingent claim with payoff at time T given by $\max[S_T - K, 0]$.

The well known Black-Scholes (BS) formula for the price of a call on this asset is given by

$$C_{BS}\left(S_t, K, r, q, \tau; \sigma\right) = S_t e^{-q\tau} N\left(d_1\right) - K e^{-r\tau} N\left(d_1\right)$$

$$d_1 = \frac{-\ln m + \left(r - q + \frac{1}{2}\sigma^2\right)\tau}{\sigma\sqrt{\tau}} \qquad d_2 = d_1 - \sigma\sqrt{\tau}$$

where $\tau = T - t$ is the time-to-maturity, t is the current time, $m = K/S$ is the moneyness of the option, r and q are the continuously compounded risk-free rate and dividend yield and $N(\cdot)$ is the cumulative normal distribution function. Suppose a market option price, denoted by $C_M\left(S_t, K\right)$, is observed. The Black-Scholes implied volatility for this option price is that value of volatility which equates the BS model price to the market option price as follows

$$\sigma_{BS}\left(S_t, K\right) > 0$$
$$C_{BS}\left(S_t, K, r, \tau; \sigma_{BS}\left(S_t, K\right)\right) = C_M\left(S_t, K\right)$$

If the assumptions underlying the BS option pricing model were correct, the BS implied volatilities for options on the same underlying asset would be constant for different strike prices and maturities.

Many different option pricing models have been proposed as alternatives to the BS model. Examples include stochastic volatility models and jump diffusion models which allow for more complex asset price dynamics. We examine one such simple extension of the BS model known as the Variance Gamma (VG) option pricing model. The idea is to model stock price movements occurring on business time rather than on calendar time using a time transformation of a Brownian motion. The resulting model is a three parameter model where roughly speaking we can interpret the parameters as controlling volatility, skewness and kurtosis, denoted respectively as σ, θ and ν, of the underlying asset returns distribution. Closed form option pricing formulae exist under the VG model [19].

$$C_{VG}\left(S_t, K, r, \tau; \{\sigma, \nu, \theta\}\right) = S_t e^{-q\tau} \Psi\left(d\sqrt{\frac{1-c_1}{\nu}}, \left(\alpha + s\right)\sqrt{\frac{\nu}{1-c_1}}, \frac{\tau}{\nu}\right)$$
$$- K e^{-r\tau} \Psi\left(d\sqrt{\frac{1-c_2}{\nu}}, \alpha s\sqrt{\frac{\nu}{1-c_2}}, \frac{\tau}{\nu}\right)$$

where

$$d = \frac{1}{s}\left[\ln\left(\frac{S_t}{K}\right) + \left(r - q\right)\tau + \frac{\tau}{\nu}\ln\left(\frac{1-c_1}{1-c_2}\right)\right]$$
$$\alpha = \varsigma s, \qquad \varsigma = -\frac{\theta}{\sigma^2}, \qquad s = \frac{\sigma}{\sqrt{1 + \left(\frac{\theta}{\sigma}\right)^2 \frac{\nu}{2}}}$$
$$c_1 = \frac{\nu\left(\alpha + s\right)^2}{2}, \qquad c_2 = \frac{\nu\alpha^2}{2}$$

and where Ψ is defined in terms of the modified Bessel function of the second kind.

Table 1. Market BS implied volatilities and option prices for FTSE 100 index options on the 17 March 2006. The strike prices are given in the table and the other observable inputs are $S = 5999.4, \tau = \frac{35}{365}, r = 0.0452$ and $q = 0.0306$.

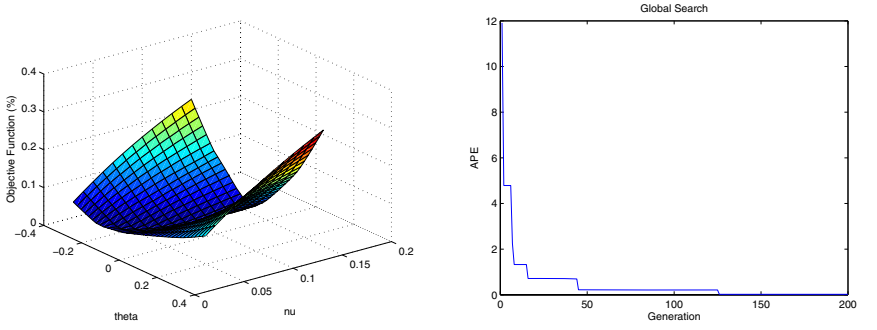
Strike price	5695.2	5845.1	5995.0	6144.9	6294.7
IV (%)	13.76	12.41	11.13	10.44	10.94
Call($)	323.67	193.63	88.67	28.03	7.99
Put ($)	12.44	31.63	75.89	164.48	293.67

4 Experimental Approach

Market makers in the options markets quote BS implied volatilities rather than option prices even though they realise BS is a flawed model. The first row in Table 1 depicts end-of-day settlement Black-Scholes implied volatilities for FTSE 100 European options on the 17 March 2006 for different strike prices and a time-to-maturity of 35 days. As can be seen the BS implied volatilities are not constant across the strike price. The second and third row in Table 1 converts the BS implied volatities into market call and put prices by substituting the BS implied volatilities into the Black-Scholes formula. The following input parameters were used to calculate the option prices, the index price is the FTSE 100 index itself $S_t = 5999.4$, the interest rate is the one month Libor rate converted into a continuously compounded rate $r = 0.0452$ and the dividend yield is a continuously compounded dividend yield downloaded from Datastream and is $q = 0.0306$. These prices are then taken to be the observed market option prices. Out-of-the money (OTM) option prices are considered most suitable for calibration purposes because of their liquidity and informational content. Hence OTM put prices were used for $K < S$ and OTM call prices were used for $K > S$ in the calibration. The calibration problem now amounts to choosing an optimum parameter vector $\Theta = \{\sigma, \nu, \theta\}$ such that an objective function $G(\Theta)$ is minimised. In this paper the objective function is chosen to be the absolute average percentage error (APE)

$$\mathrm{G}(\Theta) = \frac{1}{N} \sum_{i=1}^{N} \left| \frac{C_i - C_i(\Theta)}{C_i} \right|$$

where C_i is the observed market price on the i-th option (could be a call or a put) and $C_i(\Theta)$ is the VG model price of the i-th option with parameter vector Θ. One of the difficulties in model calibration is that the available market information may be insufficient to completely identify the parameters of a model [20]. If the model is sufficiently rich relative to the number of market prices available, a number of possible parameter vector combinations will be compatible with market prices and the objective function $G(\Theta)$ may not be convex function of Θ. A plot of the objective function versus the two parameters controlling skewness and kurtosis of the asset returns distribution, θ and ν, whilst keeping σ fixed at $\sigma = 0.1116$ is shown in figure 1(a).



(a) Objective function vs parameters (b) Objective function vs generation no.

Fig. 1. Objective function versus model parameters ν and θ and objective function versus generation number

It displays a flat profile near the minimum where many parameter combinations will yield equivalent fits. The error surface is not a straightforward error surface and a local optimiser might not converge to the true optimum. There are regions where the error surface is very flat for changes in the parameter values and there are regions where the optimiser might get not converge to global optimum.

5 Results

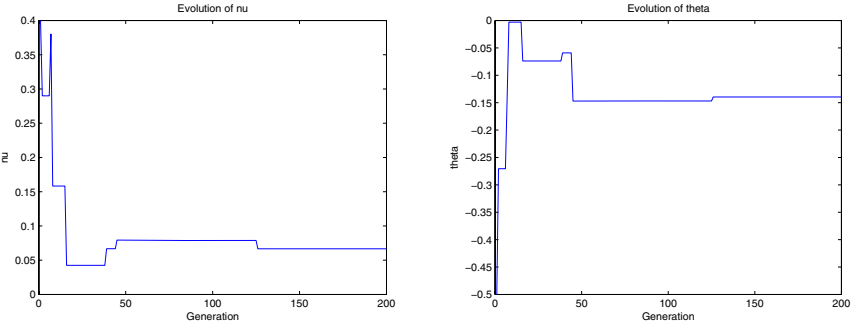
In all runs of the QIGA, a population size of 50 observed chromosomes was used, the algorithm was allowed to run for 200 generations, and all reported results are averaged over 30 runs. In order to provide a benchmark for the results obtained by the QIGA a deterministic Matlab optimiser called *fminsearch* was run 30 times with different initial parameter vectors. *Fminsearch* uses the simplex search method of [21]. This is a direct search method that does not use numerical or analytic gradients. The optimiser converged to different values for Θ for different

Table 2. Results of QIGA where the mean parameter values after 30 runs and the best performing parameter values are compared with the parameters from the matlab optimiser *fminsearch*. The resulting mean model prices from the 30 runs are compared with the market prices and the mean APE is reported.

Parameter	Mean QIGA	Best QIGA	Matlab	Market Price	Mean Model Price	Best Model Price
σ	0.0926	0.1055	0.1143	12.44	17.13	13.43
ν	0.3302	0.0234	0.0638	31.64	32.62	35.50
θ	-0.2316	-0.4258	-0.1429	75.90	65.66	83.13
				28.02	22.10	32.41
APE	2.5099	0.6000		7.99	7.03	6.75

Table 3. This table reports the QIGA objective function for different values of the exploitation and exploration parameters, respectively given by a and b

$a \setminus b$	0.4	2.0	3.6	5.2	6.8	8.4
0.05	0.5035	0.1808	0.4192	0.0411	0.0273	0.2265
0.25	0.5035	0.0791	0.4868	0.3789	0.1536	0.0992
0.45	0.5035	0.3560	0.2080	0.1984	0.0993	0.2870
0.65	0.5035	0.1298	0.1015	0.0179	0.0633	0.0907
0.85	0.5035	0.3385	0.5035	0.5035	0.5035	0.4114



(a) ν (b) θ

Fig. 2. Evolution of parameters ν and θ as a function of the generation number

initialisations of the parameter vector so the one with the optimal value for the objective function G was chosen. The results are reported in the Tables 2 and 3. As can be seen when averaged over only 30 runs the QIGA parameter vector Θ is reasonably close to the optimal parameter vector from matlab. Figure 1(b) depicts the evolution of the global objective function G (also known as APE) as a function of the generation number. Figures 2(a) and 2(b) depict the evolution of the parameters ν and θ as a function of the generation number. The results reported in Table 3 depicts the sensitivity of the objective function value to the exploitation and exploration parameters, respectively a and b. In this table a and b are varied to those values reported in while everything else remains fixed. It can be seen that the performance of the QIGA is not good when b is low regardless of what value a takes. As b increases more exploration takes place and the performance of the algorithm improves especially when a is set to intermediate values (approx. 0.65). Further sensitivity analysis would need to be conducted to find optimal values for these parameters.

6 Conclusions and Future Work

This study illustrates how a quantum-inspired evolutionary algorithm can be constructed and examines the utility of the resulting algorithm on a problem in

financial modelling known as model calibration. The results from the algorithm are shown to be robust and comparable to those of other algorithms.

Several extensions of the methodology in this study are indicated for future work. Algorithms extensions would include developing and testing a real valued QIGA and comparing its performance to the binary algorithm used in this paper. Financial applications include the calibration of more complex higher dimensional option pricing models that may contain many local minima to market data in an evolutionary setting. The use of QIGA in these types of problems may be crucial due to the potential reduction in computational time.

References

1. Spector, L. (2004). *Automatic Quantum Computer Programming: A Genetic Programming Approach*, Boston, MA: Kluwer Academic Publishers.
2. Narayanan, A. and Moore, M. (1996). Quantum-inspired genetic algorithms, *Proceedings of IEEE International Conference on Evolutionary Computation*, May 1996, pp. 61-66, IEEE Press.
3. Han, K-H. and Kim, J-H. (2002). Quantum-inspired evolutionary algorithm for a class of combinatorial optimization, *IEEE Transactions on Evolutionary Computation*, 6(6):580-593.
4. Han, K-H. and Kim, J-H. (2002). Quantum-inspired evolutionary algorithms with a new termination criterion, H_ε gate and two-phase scheme, *IEEE Transactions on Evolutionary Computation*, 8(3):156-169.
5. Yang, S., Wang, M. and Jiao, L. (2004). A genetic algorithm based on quantum chromosome, *Proceedings of IEEE International Conference on Signal Processing* (ICSP 04), 31 Aug- 4 Sept. 2004, pp. 1622-1625, IEEE Press.
6. Yang, S., Wang, M. and Jiao, L. (2004). A novel quantum evolutionary algorithm and its application, *Proceedings of IEEE Congress on Evolutionary Computation 2004* (CEC 2004), 19-23 June 2004, pp. 820-826, IEEE Press.
7. Brabazon, A. and O'Neill, M. (2006). *Biologically-inspired Algorithms for Financial Modelling*, Berlin: Springer.
8. Yang, S., Wang, M. and Jiao, L. (2004). A Quantum Particle Swarm Optimization, in *Proceedings of the Congress on Evolutionary Computation 2004*, 1:320-324, New Jersey: IEEE Press.
9. Lee C-D., Chen, Y-J., Huang, H-C., Hwang, R-C. and Yu, G-R. (2004). The non-stationary signal prediction by using quantum NN, *Proceedings of 2004 IEEE International Conference on Systems, Man and Cybernetics*, 10-13 Oct. 2002, pp. 3291-3295, IEEE Press.
10. Garavaglia, S. (2002). A quantum-inspired self-organizing map (QISOM), *Proceedings of 2002 International Joint Conference on Neural Networks* (IJCNN 2002),12-17 May 2002, pp. 1779-1784, IEEE Press.
11. Tsai, X-Y., Chen, Y-J., Huang, H-C., Chuang, S-J. and Hwang, R-C. (2005). Quantum NN vs NN in Signal Recognition, *Proceedings of the Third International Conference on Information Technology and Applications* (ICITA 05), 4-7 July 2005, pp. 308-312, IEEE Press.
12. Li, Y., Zhang, Y., Zhao, R. and Jiao, L. (2004). The immune quantum-inspired evolutionary algorithm, *Proceedings of 2004 IEEE International Conference on Systems, Man and Cybernetics*, 10-13 Oct. 2002, pp. 3301-3305, IEEE Press.

13. Jiao, L. and Li, Y. (2005). Quantum-inspired immune clonal optimization, *Proceedings of 2005 International Conference on Neural Networks and Brain* (ICNN&B 2005), 13-15 Oct. 2005, pp. 461-466, IEEE Press.
14. da Cruz, A, Vellasco, M. and Pacheco, M. (2006). Quantum-inspired evolutionary algorithm for numerical optimization, in *Proceedings of the 2006 IEEE Congress on Evolutionary Computation (CEC 2006)*, 16-21 July, Vancouver, pp. 9180-9187, IEEE Press.
15. Han, K-H. and Kim, J-H. (2003). On setting the parameters of quantum-inspired evolutionary algorithm for practical applications, *Proceedings of IEEE Congress on Evolutionary Computing* (CEC 2003), 8 Aug-12 Dec. 2003, pp. 178-184, IEEE Press.
16. Black, F. and Scholes, M. (1973). The pricing of options and corporate liabilities, *Journal of Political Economy*, 81, pp. 637-654.
17. Madan, D. and Seneta, E. (1990). The VG model for share market returns, *Journal of Business*, 63, pp. 511 - 524.
18. Madan, D. and Milne, F. (1991). Option pricing with VG martingale components, *Mathematical Finance*, 1 (4), pp. 39 - 55.
19. Madan, D., Carr, P. and Chang, E. (1998). The Variance Gamma Process and Option Pricing, *European Finance Review*, 2, pp. 79 - 105.
20. Cont, R. and Ben Hamida, S. (2005). Recovering volatility from option prices by evolutionary optimisation, *Journal of Computational Finance*, 8 (4), Summer 2005.
21. Lagarias, J.C., J. A. Reeds, M. H. Wright, and P. E. Wright. (1998) .Convergence Properties of the Nelder-Mead Simplex Method in Low Dimensions, *SIAM Journal of Optimization*, Vol. 9, Number 1, pp.112-147, 1998.

An Evolutionary Computation Approach to Scenario-Based Risk-Return Portfolio Optimization for General Risk Measures

Ronald Hochreiter

Department of Statistics and Decision Support Systems, University of Vienna,
Universitätsstraße 5/9, 1010 Vienna, Austria
ronald.hochreiter@univie.ac.at

Abstract. Due to increasing complexity and non-convexity of financial engineering problems, biologically inspired heuristic algorithms gained significant importance especially in the area of financial decision optimization. In this paper, the stochastic scenario-based risk-return portfolio optimization problem is analyzed and solved with an evolutionary computation approach. The advantage of applying this approach is the creation of a common framework for an arbitrary set of loss distribution-based risk measures, regardless of their underlying structure. Numerical results for three of the most commonly used risk measures conclude the paper.

Keywords: portfolio optimization, evolutionary computation, scenario-based financial engineering, general risk measures.

1 Introduction

Due to the increasing need for financial decision optimization algorithms for complex and non-convex optimization problems in the area of financial engineering, the amount of available biologically inspired algorithms for financial modelling increased significantly during the last years, see e.g. [1] for a recent overview of possible applications in this domain.

In this paper, we consider the well-known stochastic single-stage scenario-based risk-return portfolio optimization problem in the spirit of Markowitz [2]. To solve this problem, a standard genetic algorithm adapted from [3] is used, which is summarized in Table 1. Evolutionary computation approaches have been successfully applied to this class of portfolio optimization problems, see e.g. [4], [5], [6], [7], [8], as well as the references therein, or refer to [9].

An analysis of the proposed methods reveals that the main focus is laid on multi-criteria optimization and especially the restriction to the pure Markowitz approach, i.e. using solely the expectation vector and correlation matrix of the financial assets under consideration. To extend the application domain to general risk measures we use the stochastic scenario-based formulation of single-stage portfolio optimization. By using an evolutionary computation approach we may

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 199–207, 2007.
© Springer-Verlag Berlin Heidelberg 2007

Table 1. Meta-heuristic: Evolutionary Computation - Genetic Algorithm

```
1  P ← GenerateInitialPopulation
2  Evaluate(P)
3  while termination conditions not met do
4      P' ← Recombine(P)
5      P'' ← Mutate(P')
6      Evaluate(P'')
7      P ← Select(P ∪ P'')
8  end while
```

apply the same optimization technique and decision support framework for every loss distribution-based risk measure, regardless its underlying structure. Consider the example of Value at Risk (VaR). While VaR leads to a non-convex and non-differential risk-return optimization problem for general loss distributions, the evaluation of VaR given some specific loss distribution is simply its quantile.

The problem of estimating the correct correlation matrix, which is often mentioned and criticized, is also avoided by using the scenario-based approach.

This paper is organized as follows. Section 2 provides a summary of the basics of scenario-based portfolio optimization for general risk measures. Section 3 describes the evolutionary computation framework for this type of stochastic portfolio optimization, while Section 4 presents numerical results for commonly used yet structural heterogeneous risk measures. Section 5 concludes the paper.

2 Portfolio Optimization

We consider the classical bi-criteria portfolio optimization problem in the sense of [2]. An investor has to choose from a set of (financial) assets $\mathcal{A}$ with finite cardinality $a = |\mathcal{A}|$ to invest her available budget. The bi-criteria problem stems from the fact that the investor aims at maximizing her return while aiming at minimizing the risk at the same time. The stochastic problem definition depends on the formulation of the underlying uncertainty. Instead of using an expectation vector of size a and a correlation matrix of size $a \times a$ like the standard Markowitz approach suggests, we apply the stochastic scenario-based approach, i.e. the decision is based on a set of scenarios $\mathcal{S}$ with finite cardinality $s = |\mathcal{S}|$, where each s_i is equipped with a probability p_i with $\sum_{j=1}^{s} p_j = 1$. Each scenario contains one possible development of all a assets under consideration.

Let $x \in \mathbb{R}^a$ be some portfolio, with budget normalization, i.e. $\sum_{a \in \mathcal{A}} x_a = 1$, i.e. each x_i denotes the fraction of budget to invest into the respective asset i. We may now rewrite $\mathcal{S}$ as a matrix S to calculate the discrete Profit & Loss (P&L) distribution ℓ for some portfolio x, which is simply the cross product $\ell_x = \langle x, S \rangle$. Let $x_\rho^* \in \mathbb{R}^a$ denote the optimal portfolio given some risk measure ρ and ℓ_ρ^* denote the respective ρ-optimal discrete loss distribution.

When we reconsider the bi-criteria aspect of this portfolio optimization problem, we may subsequently map the loss distribution to these two dimensions - the return (reward, value) dimension is the expectation $\mathbb{E}(\ell_x)$ and the risk dimension

is the risk mapping $\rho(\ell_x) : \mathbb{R}^s \to \mathbb{R}$. The risk dimension received special importance due to regulatory frameworks like Basel II, as well as the academic discussion about coherence of risk measures, which has been formalized in [10]. See also [11] for a in-depth discussion on quantitative risk management and how risk measures can be used for practical risk management purposes. In this notation, a risk measure is a statistical parameter of the loss distribution ℓ_x, e.g. the Standard Deviation in the Markowitz case it, the quantile in the VaR case, or the expectation of the quantile in the Conditional Value-at-Risk (CVaR, see below) case.

Furthermore, let the set $\mathcal{X}$ denote all organizational, regulatory and physical constraints. Basic constraints, which are commonly included into $\mathcal{X}$ are:

- upper and lower limits on asset weights: $x_a \geq l, x_a \leq u \; \forall a \in \mathcal{A}$
- minimum expected profit: $\mathbb{E}(\ell_x) \geq \mu$
- maximum expected $risk : \rho(\ell_x) \leq \rho$

Often, the constraint of disallowing short selling is implicitly added, and would be modeled by setting $l = 0$ in this formulation. While such basic constraints above and their extensions can be reformulated as convex optimization problems, more involved constraints like e.g. cardinality constraints, or non-linear, non-differentiable (i.e. combination of fixed and non-linear flexible cost) transaction cost structures, as well as buy-in thresholds, or round lots lead to non-convex, non-differential models, and have motivated the application of various heuristics such as evolutionary computation techniques.

In the non-multi-criteria setting, three different main portfolio optimization formulations are commonly used. Either minimize the risk

$$\text{minimize } x : \rho(\ell_x), \text{ subject to } x \in \mathcal{X}, \tag{1}$$

or maximize the value (expectation)

$$\text{maximize } x : \mathbb{E}(\ell_x), \text{ subject to } x \in \mathcal{X}, \tag{2}$$

or apply the classical bi-criteria optimization model, where an additional risk-aversion parameter κ is defined, i.e.

$$\text{maximize } x : \mathbb{E}(\ell_x) - \kappa\rho(\ell_x), \text{ subject to } x \in \mathcal{X} \tag{3}$$

Equivalence of these three formulation has been proven for convex risk measures in [12]. We will use the third formulation (3), which is a direct reformulation our bi-criteria problem to a single-criteria problem from the view of our genetic algorithm. If formulation (1) or (2) is used, the respective constraint to limit the respectively other dimension has to be integrated to ensure the equivalence.

It should be noted, that the minimum expected profit constraint is necessary for calculating efficient frontiers, i.e. by iterating over a set of minimal expected profits, which are calculated from the scenario set $\mathcal{S}$. The critical line can also be calculated conveniently by iterating over the parameter κ.

3 Evolutionary Portfolio Optimization

We may now adapt a genetic algorithm to this stochastic portfolio optimization problem. The first issue is the selection of the genotype structure. One may either use the real-valued phenotype-equivalent or use some sort of bit-encoding. This comparison, i.e. between discrete and continuous genotypes for the portfolio selection problem has been studied in [13]. For the scenario-based framework presented below, we opted for genotype-phenotype-equivalent representation, mainly based on preliminary experiences with large scenario-sets with a huge amount of assets a, i.e. $a \geq 100$. This necessitates the need for normalization after each mutation to ensure the basic budget constraint, i.e. $\sum x_{a \in \mathcal{A}} = 1$. This can be done on the fly after each budget-constraint violating operation.

The effect of using different crossovers for the portfolio selection problem has been investigated in [14]. Three types of crossovers have been compared in that paper: discrete N-point crossovers with $N = 3$, intermediate crossovers, as well as BLX-α crossovers. For the results below, both N-point crossovers and intermediate crossovers have been used, as those two have shown to suffice the needs for solving this optimization problem.

Table 2 displays the structure of the portfolio weight chromosome and the fitness calculation. For each chromosome (portfolio) x_c its respective loss distribution ℓ_{x_c} is calculated, from which the expectation $\mathbb{E}(\ell_{x_c})$ and respective risk $\rho(\ell_{x_c})$ is computed. These two dimensions (return and risk) are mapped to a single value via a risk-aversion parameter κ, which is commonly set to 1. The aim is to maximize the fitness value, i.e. to maximize the expected return and simultaneously minimize the expected risk. The example below shows a 95% Value at Risk with an inversion of the sign, as a higher VaR equals lower risk.

Table 2. Example chromosome - fitness: κ-weighted return and risk

	Asset 1	Asset 2	...	Asset a	Return $\mathbb{E}(\ell_x)$	Risk $\rho(\ell_x)$	Fitness $\mathbb{E}(\ell_x) - \kappa\rho(\ell_x)$
$x_c =$	$(\ x_1$	x_2	...	$x_a\ )$	$\mathbb{E}(\ell_{x_c})$	$\text{VaR}_{0.95}(\ell_{x_c})$	$\mathbb{E}(\ell_{x_c}) + \text{VaR}_{0.95}(\ell_{x_c})$

The following functions have been used for the implementation of the evolutionary (genetic) algorithm:

- `c = evoGArealInitial(n, z)` - Generate random chromosome c of length n with a maximum of z zeros.
- `c = evoGArealMutationFactor(c, f, p)` - Mutate chromosome c by multiplying up to a maximum of p randomly chosen genes with factor f.
- `c1, c2 = evoGArealCrossoverIntermediate(p1, p2)` - Perform an intermediate crossover on parents p_1 and p_2 to generate children c_1 and c_2.
- `c1, c2 = evoGArealCrossoverNpoint(p1, p2, n)` - Perform an n-point crossover on parents p_1 and p_2 to generate children c_1 and c_2.
- `evoGArealNormalization(c, delta)` - Normalizes the chromosome c with $c_i = \frac{c_i}{\sum c}$. Each gene of c smaller than δ is set to 0 before normalization.

The special structure of the portfolio optimization problem is considered during the initial generation of random chromosomes in `evoGArealInitial` due to the modification of (up to) a pre-specified number of randomly chosen genes to zeros. Furthermore, the mutation `evoGArealMutationFactor` with $f = 0$ is explicitely used to create portfolio chromosomes, where some asset weights are zero. This is due to the fact, that optimal portfolios (applying commonly used risk measures) generally include a small subset of all assets under consideration.

4 Numerical Results

4.1 Implementation

The evolutionary algorithm framework has been implemented in MatLab 7.3.0 (Release 2006b). Besides the genetic algorithm functions described above, fitness selection procedures, various risk evaluations (see also Section 4.3 below), as well as an algorithm workflow engine have been developed. The code is freely available on the Web under an Academic Free License at

`http://www.compmath.net/ronald.hochreiter/`

4.2 Data

The scenario set used for further calculations consists of daily historical data of 15 Dow Jones STOXX Supersector indices (Automobiles & Parts, Banks, Basic Resources, Chemicals, Construction & Materials, Financial Services, Food & Beverage, Health Care, Industrial Goods & Service, Insurance, Media, Oil & Gas, Technology, Telecommunications, Utilities). The chosen time-frame was January 2002 to January 2006 (i.e. 4 years of data resulting in 1005 scenarios of daily index changes).

4.3 Risk Measures

Three of the most commonly applied risk measures ρ for portfolio management have been chosen to conduct a comparison, and to show the general applicability of the method presented in this paper. Standard Deviation, Value at Risk (VaR), as well as Conditional Value at Risk (CVaR) have been selected. Using the Standard Deviation for scenario-based portfolio optimization resembles the classical Markowitz approach by calculating

$$\rho = \sigma = \sqrt{\sum_{i \in \mathcal{S}} p_i (\ell_i - \mathbb{E}(\ell))^2}.$$

The Value-at-Risk at level (1-α) is the α-Quantile of the loss distribution. This risk measure gained significant importance, especially for regulatory purposes. For discrete distribution it equals the sum of the $\alpha \cdot s$ smallest values of ℓ_x. While

the Mean-VaR optimization problem is non-convex, an evaluation of the VaR value of a distribution is straightforward.

$$\rho = \mathrm{VaR}_\alpha = \inf\{l \in \mathbb{R} : \mathbb{P}(l > \ell) \leq 1 - \alpha\} = \inf\{l \in \mathbb{R} : F_l(\ell) \leq \alpha\}$$

The Conditional Value-at-Risk (CVaR), which has been used as a substitute for VaR because a linear programming reformulation is available, and this risk measure additionally exhibits the property of being a coherent risk measure. It is the expectation over the quantile (VaR) of the loss distribution, i.e.

$$\rho = \mathrm{CVaR}_\alpha = \mathbb{E}(\ell | \ell \leq \mathrm{VaR}_\alpha)$$

These three risk measures result in three different optimization program formulations, e.g. quadratic optimization in the case of Markowitz, i.e. Standard Deviation. For CVaR, a linear programming reformulation for a finite set of scenarios exists, which has been presented in [15] and [16]. Finally, there exists a variety of optimization heuristics to solve the Mean-VaR portfolio optimization problem, see especially [17] and the references therein. Of course, these reformulations are only valid if standard, convex, non-integer constraints are added, however different approaches are necessary. However, the scenario-based evolutionary algorithm framework enables a common treatment of risk measures in one coherent way.

4.4 Results

The main parameters for the evolutionary algorithm have been taken from [4] based on their results, i.e. the population size has been set to 500. A random initial population is created by setting the maximum number of zeros in `evoGArealInitial` to $a - 1$. Each new population is created by adding the 50 best chromosomes of the previous population, furthermore adding 100 children each of intermediate crossovers as well as those of 1-point crossovers from two random parents drawn out of the best 50 chromosomes. 50 set-zero mutations

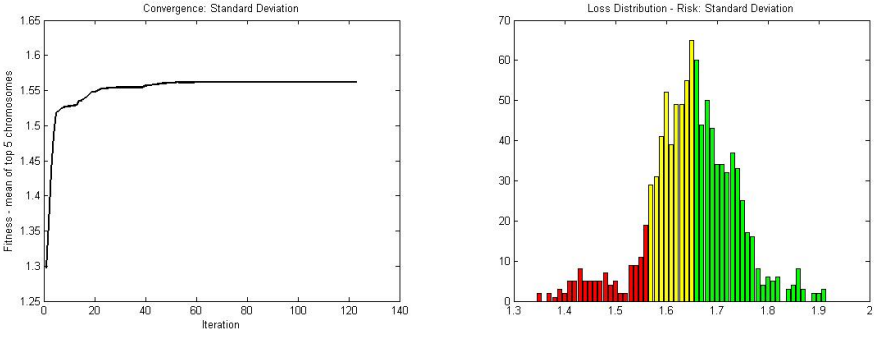


Fig. 1. Standard Deviation: Convergence and final loss distribution

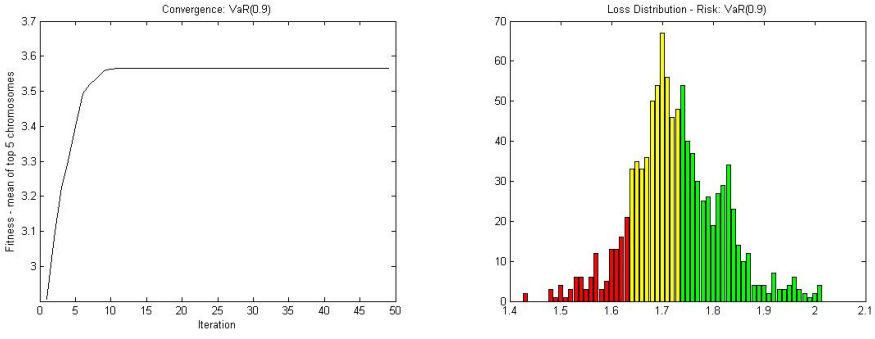


Fig. 2. VaR: Convergence and final loss distribution

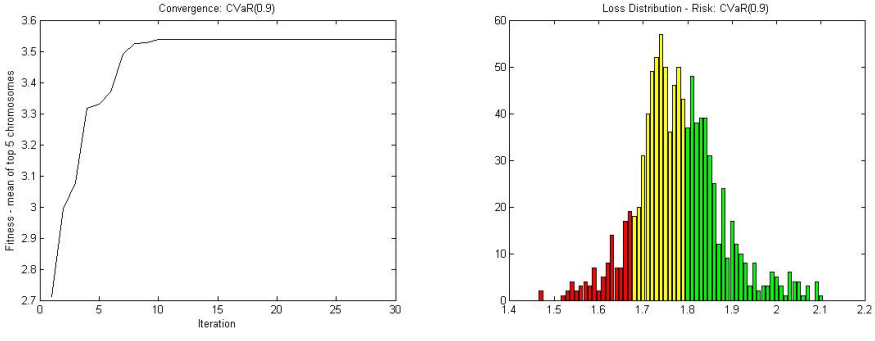


Fig. 3. CVaR: Convergence and final loss distribution

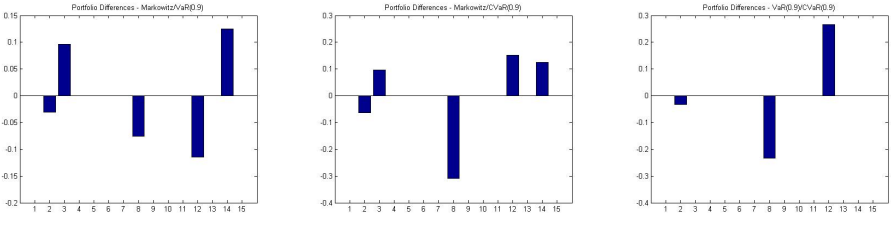


Fig. 4. Portfolio comparison: Markowitz/VaR, Markowitz/CVaR, VaR/CVaR

as well as factor mutations with $f = 2$ out of the 100 fittest were also added. Finally, 100 2-Point crossovers between one parent randomly drawn out of the 50 best and one out of the 50 worst ones have been added, as well as 100 random chromosomes, again with maximum of $a - 1$ randomly chosen zeros, to complete the offspring population. Each randomly generated or mutated chromosome, and crossover child is delta-normalized with $\delta = 0.05$. The iteration is conducted

until the difference of the mean of the fitness value of the 5 fittest chromosomes compared to this value 20 iterations before is smaller than $1e-7$.

In the case of plain risk-return optimization, the algorithm converges fast and stable, and produces expected results, as shown in Figures 1, 2, and 3 below. These figures contain convergence and the loss distribution of the final optimal portfolio respective to the chosen risk measure. The quantile α has been set to $\alpha = 0.9$ for both the VaR and the CVaR optimization. In Figure 4 a comparison of the computed optimal portfolios is shown, visualizing the differences in asset weights each between two out of the three risk measures.

5 Conclusion

In this paper, an evolutionary computation approach for solving the stochastic scenario-based risk-return portfolio optimization problem has been shown. In contrast to the previous methods mainly based on the multi-criteria aspect of the portfolio selection problem, the Markowitz approach (Correlation), or specialized heuristics for Mean-VaR optimization, this method is plainly based on the loss distribution of a scenario-set and can thus be conveniently applied to a general set of risk measures. One major advantage is the possibility to compare a set of risk measures in one coherent optimization framework, regardless their structure, which would normally necessitate the integration of linear, convex (quadratic), and global optimization or heuristic techniques.

Numerical results have been shown to validate the applicability of this approach. Furthermore, the implementation has been done from the scratch in MatLab and is available under an Academic Free License, such that new extensions can be applied without the need for a reimplementation of the basic portfolio problem.

Future research includes a study of issues, which might occur during the optimization procedure, when special non-convex constraints are added. Furthermore, a comparison between classical convex portfolio optimization approaches as well as other heuristic approaches with the scenario-based evolutionary portfolio optimization techniques presented in this paper could prove to be an additional argument to promote the use of nature-inspired algorithms for practical financial problems. While some specific issues of the portfolio optimization problem, i.e. the usually high number of asset-weights being zero, have been included in standard genetic algorithm operators, another valuable extension would be the application of e.g. hybrid mutation schemes, which are based on a preliminary analysis of the scenario set.

References

1. Brabazon, A., O'Neill, M.: Biologically inspired algorithms for financial modelling. Natural Computing Series. Springer-Verlag, Berlin (2006)
2. Markowitz, H.M.: Portfolio selection. The Journal of Finance **7**(1) (1952) 77–91

3. Blum, C., Roli, A.: Metaheuristics in combinatorial optimization: Overview and conceptual comparison. ACM Computing Surveys **35**(3) (2003) 268–308
4. Streichert, F., Ulmer, H., Zell, A.: Evolutionary algorithms and the cardinality constrained portfolio selection problem. In: Selected Papers of the International Conference on Operations Research (OR 2003), Springer (2003) 253–260
5. Schlottmann, F., Mitschele, A., Seese, D.: A multi-objective approach to integrated risk management. In Coello, C.A.C., Aguirre, A.H., E.Zitzler, eds.: Proceedings of the Evolutionary Multi-Criterion Optimization Conference (EMO 2005). Volume 3410 of Lecture Notes in Computer Science., Springer (2005) 692–706
6. Subbu, R., Bonissone, P., Eklund, N., Bollapragada, S., Chalermkraivuth, K.: Multiobjective financial portfolio design: a hybrid evolutionary approach. In: The 2005 IEEE Congress on Evolutionary Computation. Volume 2., IEEE Press (2005) 1722–1729
7. Gomez, M.A., Flores, C.X., Osorio, M.A.: Hybrid search for cardinality constrained portfolio optimization. In: GECCO '06: Proceedings of the 8th annual conference on Genetic and evolutionary computation, ACM Press (2006) 1865–1866
8. Lin, D., Li, X., Li, M.: A genetic algorithm for solving portfolio optimization problems with transaction costs and minimum transaction lots. In Wang, L., Chen, K., Ong, Y.S., eds.: Advances in Natural Computation, First International Conference, ICNC 2005, Changsha, China, August 27-29, 2005, Proceedings, Part III. Volume 3612 of Lecture Notes in Computer Science., Springer (2005) 808–811
9. Maringer, D.: Portfolio Management with Heuristic Optimization. Volume 8 of Advances in Computational Management Science. Springer (2005)
10. Artzner, P., Delbaen, F., Eber, J.M., Heath, D.: Coherent measures of risk. Mathematical Finance **9**(3) (1999) 203–228
11. McNeil, A.J., Frey, R., Embrechts, P.: Quantitative risk management. Princeton Series in Finance. Princeton University Press (2005)
12. Krokhmal, P., Palmquist, J., Uryasev, S.: Portfolio optimization with conditional value-at-risk objective and constraints. The Journal of Risk **4**(2) (2002) 11–27
13. Streichert, F., Ulmer, H., Zell, A.: Comparing discrete and continuous genotypes on the constrained portfolio selection problem. In et al., K.D., ed.: Genetic and Evolutionary Computation (GECCO 2004) - Proceedings, Part II. Volume 3103 of Lecture Notes in Computer Science., Springer (2004) 1239–1250
14. Streichert, F., Ulmer, H., Zell, A.: Evaluating a hybrid encoding and three crossover operators on the constrained portfolio selection problem. In: CEC2004. Congress on Evolutionary Computation, 2004. Volume 1., IEEE Press (2004) 932–939
15. Rockafellar, R.T., Uryasev, S.: Optimization of Conditional Value-at-Risk. The Journal of Risk **2**(3) (2000) 21–41
16. Uryasev, S.: Conditional Value-at-Risk: Optimization algorithms and applications. Financial Engineering News **14** (2000) 1–5
17. Gilli, M., Këllezi, E., Hysi, H.: A data-driven optimization heuristic for downside risk minimization. The Journal of Risk **8**(3) (2006) 1–18

Building Risk-Optimal Portfolio Using Evolutionary Strategies

Piotr Lipinski[1], Katarzyna Winczura[2], and Joanna Wojcik[2]

[1] Laboratoire des Sciences de l'Image, de l'Informatique et de la Télédétection, CNRS, Université Louis Pasteur, Strasbourg, France
lipinski@dpt-info.u-strasbg.fr
[2] Department of Mathematical Economics and e-Business, University of Information Technology and Management, Rzeszow, Poland
{kwinczura,jwojcik}@wsiz.edu.pl

Abstract. In this paper, an evolutionary approach to portfolio optimization is proposed. In the approach, various risk measures are introduced instead of the classic risk measure defined by variance. In order to build the risk-optimal portfolio, three evolutionary algorithms based on evolution strategies are proposed. Evaluations of the approach is performed on financial time series from the Warsaw Stock Exchange.

Keywords: Evolutionary Computation, Evolution Strategies, Portfolio Optimization, Risk Measures, Financial Time Series, Warsaw Stock Exchange.

1 Introduction

Evolutionary Algorithms were successfully incorporated into many fields of science and technology, among other things, into economics and finance ([4], [6], [8], [12]). This paper presents another application of Evolutionary Algorithms in this domain, namely an evolutionary approach to the problem of portfolio optimization, which consists in minimizing the risk of an investment for a desired level of expected return.

Although some analytical methods are well-known for classic versions of the problem ([1], [3]), an extension of the problem by introducing more complex risk measures and loosing several artificial assumptions requires a new efficient approach, which cannot be developed on the basis of classic methods due to the irregularity of the objective function and the search space. However, the opportunities provided by Evolutionary Algorithms ([2], [10]) may lead to an efficient optimization of portfolio structures.

Moreover, apart from theoretical constraints, which are usually considered in financial models, the approach presented focus also on a few practical constraints such as budget constraints, which means that the user of the system has only finite amount of money, as well as investor capabilities and preferences, which means that the user has to obey commonly used regulations such as paying transaction fees. Moreover, an important constraint is constituted by time restrictions and hardware limits.

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 208–217, 2007.
© Springer-Verlag Berlin Heidelberg 2007

The paper is structured in following manner: First, in Section 2, the exact problem definition is given. Section 3 describes the real-life data from the Warsaw Stock Exchange used in the computations. In Section 4, the proposed approach to the problem of portfolio optimization is presented in detail. Section 5 justifies the proposed approach presenting several benchmarks and experiments. Finally, Section 6 concludes the paper and points possible future extensions.

2 Problem Definition

In this paper, we focus on the main goal of investors, which is to optimally allocate their capital among various financial assets. Searching for an optimal portfolio of stocks, characterized by random future returns, seems to be a difficult task and is usually formalized as a risk-minimization problem under a constraint of expected portfolio return. The risk of portfolio is often measured as the variance of returns, but many other risk criteria have been proposed in the financial literature ([1]). Portfolio theory may be traced back to the Markowitz's seminal paper ([9]) and is presented in an elegant way in [3].

Consider a financial market on which n risky assets are traded. Let

$$\mathbf{R} = (R_1, R_2, \ldots, R_n)'$$

be the square-integrable random vector of random variables representing their return rates.

Denote as $\mathbf{r} = (r_1, r_2, \ldots, r_n)' \in \mathbb{R}^n$ the vector of their expected return rates

$$\mathbf{r} = (\mathbf{E}[R_1], \mathbf{E}[R_2], \ldots, \mathbf{E}[R_n])' = \mathbf{E}[\mathbf{R}]$$

and as $\mathbf{V}$ the corresponding covariance matrix which is assumed positive definite.

A portfolio is a vector $\mathbf{x} = (x_1, x_2, \ldots, x_n)' \in \mathbb{R}^n$ verifying

$$x_1 + x_2 + \ldots + x_n = 1. \tag{1}$$

Hence x_i is the proportion of capital invested in the i-th asset.

Denote as X the set of all portfolios. For each portfolio $\mathbf{x} \in X$, we define

$$R_{\mathbf{x}} = x_1 R_1 + x_2 R_2 + \ldots + x_n R_n = \mathbf{x}'\mathbf{R}$$

as the random variable representing the portfolio return rate and then

$$\mathbf{E}[R_{\mathbf{x}}] = x_1 r_1 + x_2 r_2 + \ldots + x_n r_n = \mathbf{x}'\mathbf{r}$$

is the portfolio expected return rate.

For a fixed level $e \in \mathbb{R}$ of expected return rate, let

$$X_e = \{\mathbf{x} \in X : \mathbf{E}[R_{\mathbf{x}}] = e\}$$

be the set of all portfolios leading to the desired expected return rate e. Therefore, the classic Markowitz's problem of portfolio optimization may be formulated as finding $\tilde{\mathbf{x}} \in X_e$ such that:

$$\mathbf{Var}[R_{\tilde{\mathbf{x}}}] = \min\{\mathbf{Var}[R_{\mathbf{x}}] : \mathbf{x} \in X_e\}, \tag{2}$$

where the variance is considered as the risk measure.

Such a problem, defined in the classic portfolio theory, may be solved using analytical methods ([3]). The approach has very strong mathematical foundations and completed theoretical models. In spite of this, there is a lot of competitive practical approaches, which extend these theoretical models to real investment market.

Dealing with theoretical models, strong assumptions should be fulfilled. Most of them are completely artificial and unreal such as some of the classic assumptions. Unfortunately, loosing these assumptions, the model becomes more and more complex, hence it cannot be solved in classic way.

In spite of its wide diffusion in the professional and academic worlds, the classic model is often criticized for its artificial assumptions. Although it is an interesting theoretical model, its practical applications may often misfire. Competitive portfolio optimization methods base on heuristic descended from empirical observations. Also the artificial intelligence is often used to optimize a stocks portfolio ([5], [8]).

In this paper, we extend the classic model by introducing several alternative risk measures instead of variance. The extended model cannot be solved by analytical methods, because of its complexity and lack of proper optimization tools. However, evolutionary algorithms presented in the next section can do with this problem returning satisfying results.

In order to extend the classic problem of portfolio optimization, let replace the criteria (2) by the criteria

$$\varrho(\tilde{\mathbf{x}}) = \min\{\varrho(\mathbf{x}) : \mathbf{x} \in X_e\}, \tag{3}$$

where $\varrho : \mathbf{X} \to \mathbb{R}$ is a risk measure, i.e. a function which assign to each portfolio $\mathbf{x} \in X$ its risk $\varrho(\mathbf{x}) \in \mathbb{R}$. For instance, it may represent semivariance of the return rate

$$\varrho(\mathbf{x}) = \mathbf{SVar}[\mathbf{R_x}] = \mathbf{E}[(\mathbf{R_x} - \mathbf{r_x})_-^2],$$

where

$$(\mathbf{R_x} - \mathbf{r_x})_- = \begin{cases} 0, & \text{if } \mathbf{r_x} \leq \mathbf{R_x} \\ \mathbf{R_x} - \mathbf{r_x}, & \text{if } \mathbf{R_x} < \mathbf{r_x} \end{cases},$$

the downside risk of the return rate ([1]) or other risk measure, such as these studied in ([7]).

3 Data Description

In practice, all the computations are performed on real-life data from the Warsaw Stock Exchange consisting of financial time series of price quotations of about 40 different stocks over a specific time period. On the basis of these data, the return rates are estimated. Let

$$(\xi_k^{(1)}), (\xi_k^{(2)}), \ldots, (\xi_k^{(n)})$$

denote time series representing prices of stocks $A_1, A_2, \ldots, A_n$ respectively, i.e. for each $i = 1, 2, \ldots, n$ the sequence

$$\xi_0^{(i)}, \xi_2^{(i)}, \ldots, \xi_m^{(i)}$$

contains prices of the stock A_i in consecutive time instants of the specific period and $m+1$ denotes the length of the time period (the same for all the stocks A_i). Further, let

$$r_1^{(i)}, r_2^{(i)}, \ldots, r_m^{(i)}$$

denote a time series of return rates of the stock A_i in consecutive time instants of the specific period, i.e.

$$r_j^{(i)} = \frac{\xi_j^{(i)} - \xi_{j-1}^{(i)}}{\xi_{j-1}^{(i)}}, \quad \text{for } j = 1, 2, \ldots, m.$$

Therefore, for each $i = 1, 2, \ldots, n$, the expected return rate $r_i = \mathbf{E}[R_i]$ and the variance $\mathbf{Var}[R_i]$ may be computed respectively as

$$r_i = \frac{1}{m} \sum_{j=1}^{m} r_j^{(i)}, \quad \mathbf{Var}[R_i] = \frac{1}{m-1} \sum_{j=1}^{m} (r_j^{(i)} - r_i)^2.$$

Similarly, one may compute the correlation matrix $\mathbf{\Sigma}$.

For any portfolio $\mathbf{x} \in \mathbb{R}^n$, the expected return rate $r_{\mathbf{x}} = \mathbf{E}[\mathbf{R_x}]$ and the semivariance $\mathbf{SVar}[\mathbf{R_x}]$ may be computed respectively as

$$r_{\mathbf{x}} = \sum_{i=1}^{n} x_i r_i, \quad \mathbf{SVar}[\mathbf{R_x}] = \frac{1}{m-1} \sum_{j=1}^{m} (\sum_{i=1}^{n} x_i r_j^{(i)} - r_{\mathbf{x}})^2.$$

In our approach, the problem of portfolio optimization is expressed as finding a vector $\mathbf{x} \in \mathbb{R}^n$ minimizing a given risk measure $\varrho : \mathbf{X} \to \mathbb{R}$ under the constraint that the expected return rate $r_{\mathbf{x}}$ is no lower than a given value $e \in \mathbb{R}$ (the value e is often given as the expected return rate of a specific initial portfolio $\mathbf{x}_0$). It is considered in the context of a given set of stocks $A_1, A_2, \ldots, A_n$ with time series of its prices over a given time period. Such a problem with irregular risk measures constitutes hard optimization problem and is studied with evolutionary algorithms.

4 Evolutionary Algorithms

In our research, we focus on solving the optimization problem defined in the previous section using evolution strategies and their modifications. Three different algorithms are applied: a simple evolution strategy with the famous Rechenberg's *1/5 success rule* (called **ES1**) ([2], [10]), a classic $ES(\mu, \lambda)$ evolution strategy with mutation parameters encoded in individuals (called **ES2**) ([2], [10]) and a more advanced $ES(\mu, \lambda, \varrho, \kappa)$ evolution strategy with mutation by multidimensional rotations (called **ES3**) ([11]). Modifications in these algorithms concern fitness evaluation, some restrictions in recombination and choosing the next population.

4.1 Search Space and Objective Function

In all the algorithms, the portfolio is encoded as an n-dimensional real number vector, where n is the number of stocks in the portfolio under consideration. The search space is the entire n-dimensional real number space $\mathbb{R}^n$ – although some elements of the space may not represent portfolios, they may be normalized to fulfill the condition (1). The objective function is basically given by the risk measure $\varrho : \mathbf{X} \to \mathbb{R}$, but is slightly modified by some heuristic additional factors, which are also considered by certain financial experts and stock market analysts, such as the β coefficients of the portfolio evaluated and the specific initial portfolio. Therefore, the following objective function is studied

$$F(\mathbf{x}) = \frac{1}{1 + \varepsilon_1 \cdot \varrho(\mathbf{x}) + \varepsilon_2 \cdot |\beta_{\mathbf{x}} - \beta_{\mathbf{x}_0}| + \varepsilon_3 \cdot \mathbf{Cov}(R_{\mathbf{x}}, R_{\mathbf{i}})},$$

where $\mathbf{x}_0$ denotes the specific initial portfolio, $R_{\mathbf{i}}$ denotes the return rate of the stock market index and $\beta_{\mathbf{x}}, \beta_{\mathbf{x}_0}$ denote the β coefficients of the portfolio evaluated $\mathbf{x}$ and the specific initial portfolio $\mathbf{x}_0$ respectively. Factors $\varepsilon_1, \varepsilon_2, \varepsilon_3$ are used to tune the algorithm and to adjust the importance of each component of the objective function.

These objective functions refer to some heuristics using parameters such as the β coefficient. By introducing the difference between the β_x of the generated portfolio and the β_{x_0} of the portfolio of reference, we penalize the portfolio having β_x far away from β_{x_0} of the reference. Nevertheless, the performance of a solution is defined in terms of expected return and risk of the portfolio over a test period as was mentioned in previous sections.

4.2 Algorithms Initialization

There are used several methods of generating an initial population. The simplest method is random generating with uniform probability. It consists of μ-times random choosing of an individual from the search space, where μ denote the population size. The second method uses an initial portfolio $\mathbf{x}_0$ given by the user. An initial population is chosen from the neighborhood of the given portfolio. It is done by generating a population of random modifications of the initial portfolio.

Every individual in the initial population has to fulfill the financial constraints. Thus, after random generation, every individual undergoes a validation process. If it is not accepted, it is fixed or replaced with other random generated individual. Therefore, the initial population is always correct, which means that fulfills all the desired conditions.

4.3 Evolutionary Operators

In the algorithm, common evolution operators such as reproduction and replacement are used.

In the process of reproduction, a population of size μ generates λ descendants. Each descendant is created from 4 ancestors. Reproduction consists of three parts: parent selection, recombination and mutation, repeated λ times.

Parent selection consists of choosing 4 parents from a population of size μ using one of the most popular methods, so-called "roulette wheel", where the probability of choosing an individual is proportional to its value of the objective function.

Recombination consists of generating one descendant from the 4 parents chosen earlier. It is done by one of two operators chosen randomly with equal probability: either the global intermediary recombination or the local intermediary recombination. In the first operator, genes of the descendant are arithmetic averages of genes of all the 4 parents chosen earlier. In the second operator, 2 parents are chosen from these 4 parents chosen earlier, for each gene separately, and next, the gene of the descendant is the arithmetic average of genes of the 2 parents.

After recombination, a mutation operator is applied. It depends on the algorithm. In the **ES1** algorithm, mutation is controlled by the Rechenberg's *1/5 success rule*: a random noise is added to each gene of the descendant, it is drawn with gaussian distribution $\mathcal{N}(0, \sigma)$, and the parameter σ is increased in each iteration of the algorithm when, in 5 last iterations, the number of mutations leading to improvement of individuals exceeded 20% of total mutations and decreased otherwise. The amount of increase and decrease is fixed.

In the **ES2** algorithm, mutation is controlled by a parameter σ, different for each individual, encoded in an additional chromosome in each individual. The parameter σ is an n-dimensional real number vector, in which each coordinate defines the standard deviation of the gaussian distribution of random noise for each gene of the individual. The parameter σ undergoes the evolution as well.

In the **ES3** algorithm, mutation is more complex. It is controlled by two parameters σ and α, different for each individual, encoded in two additional chromosomes in each individual. These parametres are used to drawn a random direction in the n-dimensional real number space and a random movement in this direction. Details of the mutation may be found in [11].

After mutation, each descendant must undergo a process of verification in order to satisfy the constraints as in the case of generating individuals during the algorithm initialization.

Finally, in the replacement process, a new population of size μ is chosen from the old population of size μ and its λ descendants via the deterministic selection. In the **ES3** algorithm, there are an additional constraint that each individual can survive no more than κ generations in the history.

4.4 Termination Criteria

Termination criteria include several conditions. The first condition is defined by the acceptable level of evaluation function value. The second is based on the homogeneity of population, defined as the minimal difference between the best and the worst portfolio. The third condition is defined as the maximal number of generations. The algorithm stops when one of them is satisfied.

5 Validation of the Approach

In order to validate our approach, a large number of experiments on various financial time series from the Warsaw Stock Exchange were performed. Each financial time series included daily quotations of a given stock.

Each experiment began with choosing stocks $A_1, A_2, \ldots, A_n$ constituting financial instruments available for an investor. Normally, $n = 10$ stocks were randomly chosen among all the stocks in our financial database (consisting of about 40 stocks from the Warsaw Stock Exchange). Next, an initial portfolio $\mathbf{x}_0$ was drawn corresponding to partitioning the investor's capital among the available stocks. Afterwards, a time instant t was chosen and the evolutionary algorithms presented in the previous section was applied to optimise the initial portfolio at the time t, i.e. to find an optimal portfolio $\mathbf{x}$ of equal or higher expected return rate $\mathbf{E}[\mathbf{x}] \geq \mathbf{E}[\mathbf{x}_0]$ and minimum risk measure $\varrho(\mathbf{x})$. All the computations concerning estimation of return rates were done over the period preceding the time instant t, i.e. over the time period $(t - \Delta t, t)$, where usually $\Delta t = 25$.

In each experiment, a few issues were investigated. First, the risk $\varrho(\mathbf{x}_0)$ of the initial portfolio $\mathbf{x}_0$ were compared with the risk $\varrho(\mathbf{x})$ of the built optimal portfolio $\mathbf{x}$ and the risk $\varrho(\mathbf{x}_*)$ of the portfolio $\mathbf{x}_*$ optimal according to the Markowitz model. Naturally, the built portfolio had always lower risk than the initial one. What is interesting, the built portfolio $\mathbf{x}$ had always lower risk than the portfolio $\mathbf{x}_*$, which proved that a portfolio optimal according to variance is not optimal according other risk measures. Second, the three evolutionary algorithms were compared according to the computing time and the quality of solutions. Finally, actual return rates of all the three portfolios, namely $\mathbf{x}_0, \mathbf{x}, \mathbf{x}_*$, were computed over a future time period $(t, t + \Delta t)$, where usually $\Delta t = 25$, with 5 portfolio restructurisations, each after 5 time instants, and compared. Naturally, the stock prices over the future time period were unknown during the process of portfolio optimization. Return rates were compared also with return rates of the so-called Buy&Hold strategy and the stock market index (the Buy&Hold strategy corresponds to investing all the capital at the beginning of the test period and holding to the end of it).

Table 1 shows a summary of the experiments addressing the first issue - risk comparison. Each experiment was repeated 30 times with different parameters. One may see that the risk of the optimised portfolio $\mathbf{x}$ is significantly lower than the risk of the initial portfolio as well as the minimum variance portfolio.

Table 2 presents performances of the three evolutionary algorithms. For each algorithm, a number η of experiments, when the optimum found by the algorithm was better than the optima found by the other two, is shown in the second column. In the third column, the average computing time, for each algorithm, is shown. Naturally, the computing time depends on parameters of the algorithm, which varied in experiments, but due to size constraints of the paper, they are only summarized. Not surprisingly, the last algorithm has the best performance, but it also requires the longer computing time.

Table 3 presents the comparison of return rates of the three portfolios, $\mathbf{x}_0, \mathbf{x}, \mathbf{x}_*$, over a future time period $(t, t + \Delta t)$. In the first, second and third column, there is

Table 1. Risk comparison for the initial portfolio $\mathbf{x}_0$, the built optimal portfolio $\mathbf{x}$, and the portfolio $\mathbf{x}_*$ optimal according to the Markowitz model

n	$\varrho(\mathbf{x}_0)$	$\varrho(\mathbf{x})$	$\varrho(\mathbf{x}_*)$
10	0.9322	0.3743	0.5362
10	0.2873	0.1983	0.3108
10	0.8642	0.6134	0.7134
10	0.8734	0.5654	0.6154
20	0.5481	0.3270	0.4642
20	0.8135	0.6141	0.8242
20	0.7135	0.4035	0.5193
20	0.6135	0.3985	0.4792

Table 2. Performance (a number of experiments, when the optimum found by the algorithm was better than the optima found by the others) and computing time for the three algorithms

Algorithm	η	Computing Time
ES1	57	38 s
ES2	75	53 s
ES3	108	97 s

Table 3. The number of experiments where the initial portfolio $\mathbf{x}_0$, the built optimal portfolio $\mathbf{x}$, and the portfolio $\mathbf{x}_*$ optimal according to the Markowitz model outperformed the others as well as the number of experiments where the portfolio $\mathbf{x}$ outperformed the Buy&Hold strategy and the stock market index

$\mathbf{x}_0$	$\mathbf{x}$	$\mathbf{x}_*$	B&H	Index
0	28	2	30	4
3	23	4	27	3
2	21	7	28	3
0	30	0	30	7
1	27	2	29	5
0	29	1	30	6
1	26	3	29	5
2	24	4	28	4

the number of experiments where the portfolio $\mathbf{x}_0, \mathbf{x}, \mathbf{x}_*$, respectively, turned out to outperform the others. In the next two columns, there are the number of experiments where the portfolio $\mathbf{x}$ outperformed the Buy&Hold strategy and the stock market index, respectively. One may see that the profit obtained by the evolutionary built portfolio and the minimum variance portfolio is usually slightly higher than the profit of the initial portfolio. In order to study that, further research is necessary. However, these results concern more economy and finance than evolutionary computation, so are not discussed in detail in this paper.

6 Conclusions and Perspectives

In this paper, a new approach to portfolio optimization was proposed. It rejects some assumptions used in theoretical models, introduces transaction costs and alternative risk measures such as the semivariance and the downside risk. The approach has been evaluated and validated using real data from the Warsaw Stock Exchange.

In order to evaluate this approach, the obtained investment strategy has been compared with the Buy&Hold strategy and the stock market index. To reduce the time period bias on performance, several time series have been selected. The results have demonstrated that the evolutionary approach is capable of investing more efficiently than the simple Buy&Hold strategy and the stock market index in some cases.

The presented approach can be still improved by modifying evolutionary operators, especially recombination. The fitness function study can also increase the efficiency of the method. Additional effort should be put on methods of portfolio validation in order to eliminate unacceptable solutions at the moment of its creation.

The evolutionary approach in stock trading is still in an experimentation phase. Further research is needed, not only to build a solid theoretical foundation in knowledge discovery applied to financial time series, but also to implement an efficient validation model for real data. The presented approach seems to constitute a practical alternative to classical theoretical models.

References

1. Aftalion, F., Poncet, P., Les Techniques de Mesure de Performance, Economica, 2003.
2. Back, T., Evolutionary Algorithms in Theory and Practice, Oxford University Press, New York, 1995.
3. Huang, C. F., Litzenberger, R., Foundations for Financial Economics, North-Holland, 1988.
4. Korczak, J., Lipinski, P., Evolutionary Building of Stock Trading Experts in a Real-Time System, [in] Proceedings of the 2004 Congress on Evolutionary Computation, CEC 2004, Portland, USA, 2004, pp.940-947.
5. Korczak, J., Lipinski, P., Roger, P., Evolution Strategy in Portfolio Optimization, [in] Artificial Evolution, ed. P. Collet, Lecture Notes in Computer Science, vol. 2310, Springer, 2002, pp.156-167.
6. Korczak, J., Roger, P., Stock Timing using Genetic Algorithms, [in] Applied Stochastic Models in Business and Industry, 2002, pp.121-134.
7. Lipinski, P., Korczak, J., Performance Measures in an Evolutionary Stock Trading Expert System, [in] Proceedings of the International Conference on Computational Science, ICCS 2004, ed. M. Bubak, G. van Albada, P. Sloot, J. Dongarra, Lecture Notes in Computer Science, vol. 3039, Springer, 2004, pp.835-842.
8. Loraschi, A., Tettamanzi, A. G. B., An Evolutionary Algorithm for Portfolio Selection Within a Downside Risk Framework, [in] Forecasting Financial Markets, Wiley.

9. Markowitz, H., Portfolio Selection, [in] Journal of Finance, 7, 1952, pp.77-91.
10. Schwefel, H.-P., Evolution and Optimum Seeking, John Wiley and Sons, Chichester, 1995.
11. Schwefel, H.-P., Rudolph, G., Contemporary Evolution Strategies, [in] Advances in Artificial Life, Springer, Berlin, 1995, pp.893-907.
12. Tsang, E. P. K., Li, J., Markose, S., Er, H., Salhi, A., Iori, G., EDDIE In Financial Decision Making, [in] Journal of Management and Economics, vol.4, no.4, 2000.

Comparison of Evolutionary Techniques for Value-at-Risk Calculation

Gonul Uludag[1], A. Sima Uyar[1], Kerem Senel[2], and Hasan Dag[2]

[1] Istanbul Technical University
[2] Isik University
uludag@be.itu.edu.tr, etaner@itu.edu.tr,
{ksenel,dag}@isikun.edu.tr

Abstract. The **V**alue-**a**t-**R**isk **(VaR)** approach has been used for measuring and controlling the market risks in financial institutions. Studies show that the t-distribution is more suited to representing the financial asset returns in VaR calculations than the commonly used normal distribution. The frequency of extremely positive or extremely negative financial asset returns is higher than that is suggested by normal distribution. Such a leptokurtic distribution can better be approximated by a t-distribution. The aim of this study is to asses the performance of a real coded Genetic Algorithm (GA) with Evolutionary Strategies (ES) approach for Maximum Likelihood (ML) parameter estimation. Using Monte Carlo (MC) simulations, we compare the test results of VaR simulations using the t-distribution, whose optimal parameters are generated by the Evolutionary Algorithms (EAs), to that of the normal distribution. It turns out that the VaR figures calculated with the assumption of normal distribution significantly understate the VaR figures computed from the actual historical distribution at high confidence levels. On the other hand, for the same confidence levels, the VaR figures calculated with the assumption of t-distribution are very close to the results found using the actual historical distribution. Finally, in order to speed up the MC simulation technique, which is not commonly preferred in financial applications due to its time consuming algorithm, we implement a parallel version of it.

Keywords: Value-at-Risk, Evolutionary Algorithm, Genetic Algorithm, Evolutionary Strategies, Maximum Likelihood Estimation, t-distribution, Monte Carlo Simulation.

1 Introduction

Value at Risk (VaR) is a topic, which has recently become very popular among risk managers and regulators due to the promise it holds for improving risk management. VaR gives an upper bound for the money to be lost for a given probability, usually taken as 90%, 95%, 99% or 99.9%. The traditional way is to assume that the distribution of the returns to be normal or log-normal. However, in practice this assumption seldom holds, because the tails are thicker than that

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 218–227, 2007.
© Springer-Verlag Berlin Heidelberg 2007

of a normal distribution [11]. One possible alternative is to use the Student's t-distribution, which has fat tails. The degrees of freedom of the t-distribution does not need to be integer. The Student's t-distribution offers a very tractable distribution that accommodates fat-tails. As its degrees of freedom increases, the t-distribution also converges to the normal, so we can regard the t-distribution as a generalization of the normal distribution, which usually has fatter tails [1].

Parameters of the t-distribution can be estimated through Maximum Likelihood (ML) estimation. In this study, the Log-Likelihood maximization is achieved through two different variants of Evolutionary Algorithms (EAs). These are Genetic Algorithms (GAs) and Evolutionary Strategies (ES). The result obtained using these two methods are compared. In a previous study, the ML estimation calculation was done using a real coded GA [7]. Those experiments are repeated here. Additionally, the same calculations are done using an ES [6] approach in this study. The results show that both EA techniques give a similar solution quality. However, as expected, due to its faster convergence properties, the ES used in this study requires significantly fewer fitness evaluations to achieve the same solution quality than the GA.

In this study, 8 different shares' closing prices from the Istanbul Stock Exchange (ISE) between 01.01.1994 and 25.11.2005 are used [10]. These shares are: ADNAC, ALARK, DOHOL, EREGL, FROTO, ISCTR, KCHOL, and VESTL.

The paper is organized as follows: Section 2 introduces the VaR measure and provides an overview of the problem. In section 3, the ML technique is explained. Section 4, gives a brief look into the EAs used. In section 5, the experimental design is outlined. Section 6, gives the experimental results and finally section 7, concludes the paper and provides possible extensions to the current study.

2 Value-at-Risk (VaR)

Recent proposals for the disclosure of financial risk call for firm-wide measures of risk. A standard benchmark is the Value-at-Risk (VaR). For a given time horizon t and a confidence level p, the VaR is the loss in market value over the time horizon t that is exceeded with probability $1 - p$ [3].

The Bank for International Settlements (BIS) has set p to 99-percent and t to 10-working days for the purpose of measuring the adequacy of bank capital, while JPMorgan has set p to 95-percent level and t to 1-working day. The VaR definition is based on two fundamental elements; holding period (1-/10-working day) and confidence level (95% or 99%). The following subsections describe the VaR techniques [4] used in our study.

2.1 Parametric VaR Method

Parametric techniques involve the selection of a distribution for the returns of the market variables, and the estimation of the statistical parameters of these returns [2]. Two parametric VaR methods, which are used in this study, are explained in section 5.2.

2.2 Historical Simulation Method

This approach can be seen as a simplified Monte Carlo (MC) simulation method. In this model, historical data are used to produce scenarios. Therefore, the assumption of normality and the computation of volatility and correlations are not required.

In this model, historical VaR is computed for comparing with the results obtained with the other assumptions [8].

2.3 Monte Carlo (MC) Simulation Method

Computation of VaR under MC simulation includes four steps: first, volatilities and correlations among risk factors are computed; second, expected price/rates under the chosen distribution using computed volatilities are produced; third, random expected prices are produced; finally, the value of a portfolio using the computed prices are calculated [8]. In this study one share's closing prices are used in experiments.

3 Maximum Likelihood (ML) Estimation

In this study, the t-distribution parameters are estimated with ML estimation. To perform the ML estimation [12] for the t-distribution, the following steps are done. Start with the *pdf* (probability density function) of the t-distribution given as:

$$f(r;\nu,\mu,\gamma) = \frac{\Gamma(\frac{\nu+1}{2})}{\Gamma(\frac{\nu}{2})\gamma\sqrt{\pi\nu}(1+(\frac{r-\mu}{\gamma\sqrt{\nu}})^2)^{\frac{(\nu+1)}{2}}} \tag{1}$$

where r is a logarithmic return values of the shares, μ is a location parameter, γ is a scale parameter, and ν is a shape (degrees of freedom) parameter. The standard t-distribution assumes $\mu = 0$, $\gamma = 1$ and ν to be a real number (e.g., $\nu \cong 3.5$). The log-likelihood function is obtained as

$$\wedge = N[\log\Gamma(\frac{\nu+1}{2}) - \log\Gamma(\frac{\nu}{2}) - \frac{1}{2}\log\pi\nu - \log\gamma] - \frac{\nu+1}{2}\sum_{i=1}^{N}\log(1+(\frac{r_i-\mu}{\gamma\sqrt{\nu}})^2) \tag{2}$$

Unlike for the normal distribution, no analytical expressions are available for the maximum log-likelihood estimates of μ, γ and ν [9]. Thus numerical techniques need be used. Due to this, in this study, EA approaches are used.

4 Evolutionary Algorithms (EA)

Among the set of population-based search and optimization heuristics, the development of EA [5] has been very important in the last decade. EAs are used successfully in many applications of high complexity.

"Evolutionary Algorithms" [5] is a term that covers a group of heuristic approaches to problem solving using models of natural mechanisms and principles based mainly on Darwin's theory of evolution and the Mendelian principles of classical genetics. EAs work on a population of individuals, each of which represents a solution to the problem and they use an iterative, stochastic search process which is guided based on the goodness or badness of current solutions.

The basic variants of EAs are GAs, Genetic Programming, Evolutionary Programming and ES [5]. For the estimation of the t-distribution's parameters, a real coded GA and an ES are appropriate. Detailed explanation of the GA and ES used in this study are given in the next section.

5 Experimental Design

The scope of this study covers the comparison of two EA variants (GAs and ES) for parameter estimation, which is used in VaR computation.

In this study there are 2836 data points (asset values) for each share. For this study, GNU Scientific Library (GSL) is used for generating all random numbers [13]. The experimental design is given in detail below.

5.1 Setting Up the EA

In this study two EA techniques are performed: ES and GA. The performance of these approaches is evaluated through a series of tests.

The objective function of the EAs can be given as below:

$$\begin{aligned} max : \Lambda(\nu, \mu, \gamma) \\ s.t. \\ 1 \leq \nu \leq 5 \\ 0.0001 \leq \mu \leq 1 \\ 0.001 \leq \gamma \leq 2 \end{aligned} \tag{3}$$

In this study chromosomes represent a set of t-distribution parameters, which will be used for different VaR calculations. Based on our preliminary experiments, these parameters are real numbers in the range as given in Eq. 3. The chromosome consists of the three parameters of the t-distribution which are μ, γ and ν.

The fitness of an individual is obtained by calculating the log-likelihood value based on the parameters given by the chromosome using Eq. 2. The assumptions for the ranges of parameters follow from a similar work [9]. In this study, standard implementations for each of the EAs are used. The general algorithmic flows of the GA and the EA are given below.

```
GA:
generate initial population randomly
evaluate initial population
repeat
    select pairs
    perform recombination
        perform mutation
        evaluate population
        do elitism
until endOfGenerations

ES:
generate initial population randomly repeat
    repeat lambda times
        select 2 parents randomly
        perform recombination
        perform mutation
        evaluate population
    end
    perform survivor selection
until endOfGenerations
```

Table 1. Description of the EAs for the maximization of log-likelihood function

	Genetic Algorithm	Evolutionary Strategies
Representation	Floating point	Floating point
Parent Selection	Tournament Selection ts=2	N/A
Recombination	Uniform Cross-over	Discrete & Intermediary Cross-over
Crossover Probability	0.8	0.8
Mutation	Gauss Mutation	Self Adaptive Gauss Mutation
Mutation Probability	100%	100%
Survival Selection	Generational	$(\mu + \lambda)$
Number of Generations	5000	5000
Population Size	100	$\mu = 20, \lambda = 100$
Chromosome Size	3	3
Number of Runs	20	20
Initialization	Random	Random
Elitism	Yes	N/A

The parameters and operators chosen for EAs are given in Table. 1. The settings are taken as those most commonly used. Further experiments with better settings are being done. In this study initial populations are generated randomly. For the initial population generation and also during mutation, the lower and upper bounds for the parameters are taken into consideration. Thus if a new gene value exceeds the predetermined boundary values, mirroring is performed.

Genetic Algorithm (GA). For the GA, parent selection is performed by tournament selection ($t_s = 2$), and recombination is performed by 2-point uniform cross-over. Since the chromosome length is 3, using 2-point crossover is not very useful. For Gauss mutation, different mutation step sizes (standard deviation of the Gaussian distribution)are chosen as a results of experiments. Each parameter are assumed as given below.

$\nu \Rightarrow N(0, \sigma_\nu) \Rightarrow \sigma_\nu = 1$
$\mu \Rightarrow N(0, \sigma_\mu) \Rightarrow \sigma_\mu = 0.0001$
$\gamma \Rightarrow N(0, \sigma_\gamma) \Rightarrow \sigma_\gamma = 0.001$

These values are chosen based on the ranges of μ, γ and ν parameters. Experimenting with population size and crossover probability shows that their values do not affect the outcome significantly, thus the above settings are chosen for this study. When performing elitism, the worst individual of the current population is replaced by the previous best individual of the population.

Evolutionary Strategies (ES). For the ES, a chromosome consist of two parts: Object variables, which are the genes, and Strategy parameters, which are the mutation step sizes (deviations). These are represented also as real numbers. Recombination generates one child from the two parents. Similar to the uniform cross over used in the GA, for the object variables of the chromosome, discrete recombination, which selects one of the parent values chosen equal randomly, is used with the same P_c. For strategy parameters, recombination is performed as intermediary recombination, which averages parental values.

In this study, for the ES, self adaptive gauss mutation is performed. Self adaptive parameter τ is chosen as $1/\sqrt{2\sqrt{n}}$, where $n = 3$ is the chromosome length.

In this study $(\mu + \lambda)$ selection (plus strategy) is performed. It acts on the set of parents and children combined. Selection is performed by ranking the fitness values of the individuals and selecting best μ of $(\mu + \lambda)$.

5.2 Applying the MC to VaR

A critical part of a MC simulation is the generation of random variables. Firstly random numbers are generated from the standard form of distributions. For this study, GSL Random Number Generator Library is used to generate the pseudo random numbers [13]. In this study the MC for VaR is computed both with the normal and the t-distribution assumptions for single-stock portfolios.

In this paper the degrees of freedom parameter is allowed to be non-integer. A nice property of this class of distributions is that kurtosis and degrees of freedom have a simple relationship.

Parametric VaR with Normal Distribution's Approach. The VaR associated with normally distributed log returns is [8];

$$VaR(h) = P - P^* = P - e^{\mu h + \alpha\sigma\sqrt{h} ln P} \tag{4}$$

where P is the current value of portfolio, P^* is the $(1-CL)$ percentile (or critical percentile) of the terminal value of the portfolio after a holding period of h days,

and α is the standard normal variate associated with our chosen confidence level (e.g., $\alpha = -1.645$ so if having a 95% confidence level).

Parametric VaR with Normal *t*-Distribution's Approach. The VaR associated with the t-distribution method is [8];

$$VaR(h) = P - P^* = P - e^{\mu h + \alpha_\nu \sigma \sqrt{h} ln P} \tag{5}$$

where P is the current value of portfolio, P^* is the $(1-CL)$ percentile (or critical percentile) of the terminal value of the portfolio after a holding period of h days, α_{df} is the Student's t variate corresponding to the chosen confidence level, and df is the number of degrees of freedom [8].

Normal Random Number Generator. The pseudo normal random numbers (r_n) for each MC simulation are generated as; $r_n = \mu + \sigma Z \Rightarrow Z \sim N(0,1)$ where; μ is the mean of asset returns, σ is standard deviation of the asset returns, and Z is a standard normal random number, which is generated by GSL [13].

***t* Random Number Generator.** The t random numbers (r_t) for each MC simulations are generated as; $r_t = \mu + \gamma Z \Rightarrow Z \sim T(\nu)$ where; μ is the location parameter, γ is the scale parameter, ν is the non-integer degrees of freedom, $T(\nu)$ is a number from the standard t-distribution (assumes $\mu = 0$, $\gamma = 1$). Z is a standard t-distributed random number variable with respect to ν, which is generated by GSL.

6 Experimental Results

μ, γ and ν for the eight shares data are estimated using the EAs described in the previous section. The parameters reported in Table. 2 are the best values obtained from the 20 runs of the GA and ES.

Table 2. Estimated parameter of t-distribution with EA and mean best fitness averages and standard deviations for the first hitting times

	Genetic Algorithm					Evolutionary Strategies				
Shares	ν	μ	γ	Fitness	Mean 1_{st} Success	ν	μ	γ	Fitness	Mean 1_{st} Success
ADANAC	2.87	0.71e-3	2.19e-2	5735.77	144±3.18	2.87	0.70e-3	2.20e-2	5735.80	32±1.60
ALARK	3.18	1.88e-3	2.38e-2	5641.80	145±3.57	3.16	1.73e-3	2.35e-2	5641.84	34±1.89
DOHOL	3.70	1.38e-3	3.01e-2	5114.28	143±2.98	3.67	1.59e-3	2.98e-2	5114.21	31±1.24
EREGL	3.25	1.27e-3	2.42e-2	5596.59	142±2.74	3.20	1.29e-3	2.42e-2	5596.59	33±1.17
FROTO	3.13	1.74e-3	2.40e-2	5603.31	145±3.62	3.18	1.68e-3	2.41e-2	5603.32	33±1.76
ISC	3.37	0.89e-3	2.55e-2	5468.75	144±3.02	3.27	0.10e-3	2.55e-2	5468.15	32±1.45
KCHOL	3.57	1.23e-3	2.48e-2	5618.85	142±2.81	3.55	1.17e-3	2.48e-2	5618.89	31±1.47
VESTL	2.83	1.24e-3	2.52e-2	5330.15	145±3.51	2.76	1.20e-3	2.51e-2	5330.15	34±1.78

Table 3. VaR values of 8 shares for different VaR methods

ADNAC	**90%**	**95%**	**99%**	**99.9%**
RealVaR	**3.31**	**4.90**	**10.00**	**13.75**
MC-*t* (GA)	3.34	4.92	10.06	15.75
MC-*t* (ES)	3.37	4.95	10.03	14.62
Pr-*t* (GA)	3.23	4.49	7.81	14.48
Pr-*t* (ES)	3.26	4.53	7.88	14.60
MC-Nr	4.10	5.10	7.77	10.62
Pr-Nr	4.20	5.40	7.62	10.05

ALARK	**90%**	**95%**	**99%**	**99.9%**
RealVaR	**3.48**	**5.17**	**10.41**	**14.42**
MC-*t* (GA)	3.46	5.15	10.38	14.59
MC-*t* (ES)	3.46	5.18	10.40	15.04
Pr-*t* (GA)	3.40	4.77	8.36	15.53
Pr-*t* (ES)	3.37	4.72	8.27	15.37
MC-Nr	4.21	5.33	8.11	11.05
Pr-Nr	4.27	5.49	7.75	10.22

DOHOL	**90%**	**95%**	**99%**	**99.9%**
RealVaR	**4.40**	**6.41**	**10.97**	**17.96**
MC-*t* (GA)	4.38	6.43	10.95	17.93
MC-*t* (ES)	4.40	6.40	10.90	17.57
Pr-*t* (GA)	4.38	6.08	10.54	19.31
Pr-*t* (ES)	4.31	6.00	10.42	19.12
MC-Nr	4.93	6.53	9.57	12.61
Pr-Nr	5.10	6.55	9.19	12.07

EREGL	**90%**	**95%**	**99%**	**99.9%**
RealVaR	**3.59**	**5.26**	**10.04**	**13.54**
MC-*t* (GA)	3.57	5.26	10.04	15.17
MC-*t* (ES)	3.58	5.26	9.99	13.40
Pr-*t* (GA)	3.52	4.90	8.55	15.83
Pr-*t* (ES)	3.52	4.90	8.55	15.83
MC-Nr	4.37	4.53	8.14	11.09
Pr-Nr	4.37	5.62	7.93	10.45

FROTO	**90%**	**95%**	**99%**	**99.9%**
RealVaR	**3.44**	**5.18**	**10.31**	**14.84**
MC-*t* (GA)	3.43	5.18	10.30	16.33
MC-*t* (ES)	3.45	5.18	10.30	15.61
Pr-*t* (GA)	3.45	4.83	8.44	15.67
Pr-*t* (ES)	3.46	4.85	8.48	15.73
MC-Nr	4.17	5.58	8.21	11.49
Pr-Nr	3.35	5.61	7.92	10.44

ISCTR	**90%**	**95%**	**99%**	**99.9%**
RealVaR	**3.73**	**5.05**	**10.18**	**14.29**
MC-*t* (GA)	3.74	5.19	10.18	16.12
MC-*t* (ES)	3.79	5.19	10.14	15.45
Pr-*t* (GA)	3.75	5.21	9.03	16.64
Pr-*t* (ES)	3.83	5.29	9.11	16.71
MC-Nr	4.63	5.94	8.92	12.10
Pr-Nr	4.81	6.19	8.73	11.49

KCHOL	**90%**	**95%**	**99%**	**99.9%**
RealVaR	**3.59**	**5.23**	**9.41**	**14.10**
MC-*t* (GA)	3.59	5.20	9.43	15.14
MC-*t* (ES)	3.59	5.23	9.40	14.56
Pr-*t* (GA)	3.61	5.04	8.76	16.20
Pr-*t* (ES)	3.61	5.04	8.76	16.20
MC-Nr	4.09	5.43	8.01	10.96
Pr-Nr	4.28	5.51	7.77	10.23

VESTL	**90%**	**95%**	**99%**	**99.9%**
RealVaR	**3.96**	**5.76**	**10.87**	**17.47**
MC-*t* (GA)	4.01	5.76	10.87	18.35
MC-*t* (ES)	3.97	5.75	10.83	18.64
Pr-*t* (GA)	3.67	5.12	8.90	16.44
Pr-*t* (ES)	3.66	5.10	8.87	16.38
MC-Nr	4.75	6.19	9.12	12.63
Pr-Nr	4.92	6.32	8.89	11.69

It is possible to use the best values over all the runs for the VaR calculations since the standard deviations of the mean best finesses are very low [1]. As can be seen in Table. 2, the results produced by the GA and the ES are very similar. Table. 2 shows that the ES requires much fewer fitness evaluations than the GA to achieve the same solution qualities

[1] Standard deviation values are not shown here because with a 2-digit after the comma precision, the standard deviation values are small.

In this study, the estimated parameters of the t-distribution are applied to the MC simulations and results of the different VaR method implementations are used. The MC simulation results, which are given in Table. 3, are taken over 5000 simulations and these are the mean values obtained from 100 runs [2]. The MC versions' standard deviation values (γ_{MC}) are zero.

The results are given in Table. 3. In all sub-tables, "$RealVaR$" is computed using the Historical VaR method, "MC-t (GA)" is computed using MC simulation method for t-distribution's approach with the GA, "MC-t (ES)" is computed using MC simulation method for t-distribution's approach with the ES, "Pr-t (GA)" are computed using the Parametric VaR with t-distribution's approach with the GA, "Pr-t (ES)" are computed using the Parametric VaR with t-distribution's approach with the ES, the "Pr-Nr" is computed using the Parametric VaR with normal distribution's approach and "MC-Nr" is computed using the MC simulation method for normal distribution.

In actual financial applications, VaR values for high confidence levels are usually preferred. As can be seen from the results, the VaR values calculated using the t-distribution whose parameters were optimized using an EA are the best in each case. This shows that the assumption of normality on the returns data is not very realistic and gives a systematic error. Since the results for the GA and the ES for each method are close, it can be said that both the GA and the the ES are suitable to be used. However, since ES requires less fitness evaluations to provide the results, it should be preferred over the GA.

MC simulation method gives good approximation to calculating the VaR, however this computation requires very long computation times. Due to this problem a parallel implementation becomes inevitable. To overcome this issue, in this study, the MC simulation method is parallelized. Since parallelization only changes the computation time and does not affect the solution quality, it remains out of the scope of this paper. Thus no further results are reported.

7 Conclusions and Future Studies

This study shows that the assumption of a t-distribution with parameters estimated through EAs is far better than the assumption of normal distribution in terms of the proximity of the corresponding results to the historical VaR. This is particularly true for higher confidence levels, which are commonly used in the financial industry. Actually, the parametric VaR figures with the assumption of t-distribution are shown to be sufficiently reliable to be used in practice. The main aim of the study is to compare a real coded GA with an ES which incorporates a self adaptive mutation mechanism for estimating the distribution parameters of the t-distribution. The results show that the solution quality for both techniques are similar, however the ES requires significantly less fitness evaluations than the GA. Thus it is more favorable to use the ES for this

[2] The number of simulations and the number of runs are chosen in order to facilitate the parallel implementation of MC simulation.

purpose. Based on the successful results obtained in this study a further extension may be the utilization of VaR for different distribution types such as the pareto distribution.

References

1. Jackson, P., Maude, D.J., Perraudin, W.: Bank Capital and Value at Risk. The Journal of Derivatives Vol. 4 (1997) 73-90
2. Engelbrecht, R.: A Comparison of Value-At-Risk Methods for Portfolios Consisting of Interest Rate Swaps and FRAs. Economics Series Working Papers, University of Oxford, Department of Economics (2003)
3. Duffie, D., Pan, J.: An Overview of Value at Risk. Journal of Derivatives Vol.4. (1997) 7-49
4. Jorion, P.: Value at Risk: The new Benchmark for Controlling Market Risk. RiskMetrics Technical Manual, McGraw-Hill, New York (1997) 847-860
5. Eiben, A.E., Smith J.E.: Introduction to Evolutionary Computing. Springer, Berlin (2003)
6. Bayer, H.G., Schwefel H.P.: Evolution Strategies a Comprehensive Introduction. Natural Comp. 1 (2002) 3-52
7. Uludag, G., Senel, K., Etaner-Uyar, A.S., Dag, H.: ML Estimation of Distribution Parameters for VaR Calculation Using Evolutionary Algorithm. WSEAS Transaction on Business and Economics (2005)
8. Dowd, K., Blake, D., Cairns, A.: Long-Term Value at Risk. Journal of Risk Finance Vol.5 (2), (2004) 52-57
9. Van den Goorbergh, R.W.J., Vlaar, P.: Value-at-Risk Analysis of Stock Returns Historical Simulation, Variance Techniques or Tail Index Estimation?. DNB Staff Reports, Amsterdam (1999)
10. http://www.ise.org
11. http://www.gloriamundi.org
12. http://www.weibull.com/AccelTestWeb/mle_maximum_likelihood_parameter_estimation.htm
13. http://www.gnu.org/software/gsl/manual

Using Kalman-Filtered Radial Basis Function Networks to Forecast Changes in the ISEQ Index

David Edelman

University College, Dublin
david.edelman@ucd.ie

Abstract. A Kalman-Filtered Feature-space approach is taken to forecast changes in the ISEQ (Irish Stock Exchange Equity Overall) Index using the previous five days' lagged returns solely as inputs. The resulting model is tantamount to a time-varying (adaptive) technical trading rule, one which achieves an out-of-sample Sharpe ('reward-to-variability') Ratio far superior to the 'buy-and-hold' strategy and its popular 'crossing moving-average' counterparts. The approach is contrasted to Recurrent Neural Network models and with other previous attempts to combine Kalman-Filtering concepts with (more traditional) Multi-layer Perceptron Network models. The new method proposed is found to be simple to implement, and, based on preliminary results presented here, might be expected to perform well for this type of problem.

1 Background

While the full range of literature relating to attempts to gain excess returns from technical (i.e. price history) information is too diverse to summarise effectively here, that part of it having to do with Machine Learning is more limited. As mentioned in [1] and elsewhere, some of the Key papers in the area by Lebaron, Lakonishok, and Lo (see [3] and [4] in particular) have suggested the possibility of Weak form inefficiencies existing in some markets, consistent with the notion of Bounded Rationality [5], but the results obtained have stopped short of definitively asserting the affirmative result.

If, as many believe, one supposes that any efficaceous technical trading rule (if such existed) could not be constant over all time (otherwise it would be discovered and exploited to oblivion), the problem of tracking a potential time-varying rule arises. In this framework, the first method which would naturally present itself would be the Kalman Filter. Unfortunately, for the most part, this would tend to limit the class of models to Linear functions, which one would doubt as lacking the 'subtlety' which one might expect to be a property of an effective trading model.

Perhaps for this very reason, there has recently been a number of attempts to combine Kalman-Filtering concepts with nonlinear models such as Neural Networks [2], arguably with limited success. However, one approach which appears to have been overlooked is that of applying Kalman Filters to the linear output layer of a network with fixed nonlinear feature space, prototypically, one

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 228–232, 2007.
© Springer-Verlag Berlin Heidelberg 2007

based on Radial Basis Functions at fixed, pre-determined centers. It is the latter approach which will be adopted here, with what appear to be very promising preliminary results, from a very simple model.

2 Methods

In what follows, we outline what is a remarkably simple but effective method for fitting a Kalman-Filtered Radial Basis Function Network, to the problem of forecasting daily changes in the Irish ISEQ index merely from 5 lagged daily changes.

As mentioned previously, the key to applying Radial Basis Function networks in the Kalman setting is to keep the centers (and bandwidth) of the Feature Space constant over time.

Since the lags forming the inputs have mean zero and standard deviation approximately equal to .01, a set of centers formed by the $\{-.01, 0, .01\}^5$ grid might be likely to each be near the body of input points.

To this end, let x_i and c_j denote the i^{th} input and j^{th} center point. Then the Feature Space matrix $P(x; c)$ may be defined, where

$$P_{ij} = \exp(-\frac{1}{2}||x_i - c_j||^2/b^2),$$

$||\cdot||$ denoting the Euclidean distance (in 5-space), for some suitably-chosen 'bandwidth parameter' b. Thus, the basic model is that

$$y_i = P(x_i; c) \cdot w^{(i)} + \epsilon_i,$$

where the weights $w^{(i)}$ are assumed to be evolving over time in an unobservable fashion, with

$$w^{(i)} = w^{(i-1)} + \eta_i,$$
$$E(\eta_i) = 0, \quad Cov(\eta_i) = \delta^2 I_m,$$

m in this case m representing the 243-strong dimension of the chosen basis and δ a tuning 'innovation' parameter (I_m the m by m Identity matrix).

The solution to such a Kalman system is well-known and will not be formally restated here, though notionally the solution may be described as daily 'upadate' application of the following four steps (to incorporate new information):

- 1. Compute current day's forecast percentage change
- 2. Add innovation component to Covariance Estimate matrix
- 3. Use above results plus new observation for 'error-correction'
- 4. Update State Covariance Estimate matrix, given above

Of course, in practice, a system cannot be iterated from nothing, but must start somewhere.

In light of this, in order to start the system we fit the first 250 days (1 year of trading) as a group using Ordinary Least Squares, and then begin the Kalman

iteration, forecasting the one-day return, crucially *out of sample* over the remaining 1050 days.

Next, given day-by-day forecast relative changes (which are 'out-of-sample', using only previous days' data) we use a simple trading rule based on investment in proportion to the predicted return $\hat{y}_i$ [this rule is based on the solution to a series approximation of Expected Logarithmic Utility, or so-called 'Kelly' criterion]:

$$b_i = K\hat{y}_i,$$

where K times the daily variance of return is a suitably chosen fraction of current level of Wealth.

In order to evaluate the results of applying this rule to the Kalman Filter forecasts, a plot of cumulative returns which would have been achieved via the trading rule will be produced, and a Sharpe ('Reward-to-Variability') calculated, along with a 'Beta' with respect to the index. As it happens, the empirical Beta for this rule is approximately zero (as will be discussed below), and the average net position (and hence cost of carry) zero. Hence, any significant return obtained, if it could be demonstrated to be net of transaction costs would constitute a violation of the Weak form of the Efficient Markets Hypothesis. While preliminary calculations suggest that a statistically significant inefficiency exists here 'in-house' (for brokers themselves, who are not charged transaction costs) this is not the emphasis of the current paper, and hence will be deferred to a future article.

3 Results

For this study, the (logarithmic) returns for 1306 trading days (from September 2001 to September 2005) of the Irish ISEQ Index index were computed and lags produced, resulting in an input series consisting of 5 lags by 1300 observations, with a corresponding output variable of 1300 daily returns. In this case, all of the variables, input and output, have similar character, being returns series with approximate mean zero and standard deviation 1.1%.

As has been mentioned previously, the method applied here is that of Radial Basis Function Networks, where a single hidden layer is used, as well as the Gaussian Kernel. Most often with models of this type, the centers of the Basis Functions are taken to be the input datapoints themselves, thus ensuring they occur in the vicinity of the actual data. However, as the present problem requires the basis functions to remain constant and future input datapoints are here not to be known in advance, this approach cannot be taken. Instead, a 5-dimensional grid (as centers for Radial Basis Functions) is thought to constitute a sensible alternative, as each point in the input dataset can be expected to be relatively close to some point in the grid, while the grid points themselves can be expected to be fairly evenly spread throughout the data. For Radial Basis Function Networks which use input data points as centers, the bandwidth chosen always effectively decreases with sample size. In this case, however, the number of centers remains constant (at $3^5 = 243$) and the bandwidth also, here set at .01 (though sensistivity analysis performed indicates that the results are

not too sensitive to this choice, so long as the basis functions are not too highly correlated nor highly concentrated).

Formally, the specification of an initial parameter value and covariance matrix at the start of a Kalman Filter for regression parameters is equivalent to the specification of a Ridge penalty configuration in Regression, where the 'initial values' in the Kalman Filter case are merely the values towards which coefficients are shrunk in Ridge Regression. In this case, then, it was decided to apply Ridge Regression to an initial sample, taken to be 250 days, the results of which were used to initialise a Kalman Filter beginning at day 251. From this point onwards, the coefficients of the network merely follow from the standard Kalman update equations, where the only parameter which requires specification is the process innovation Covariance matrix, which is taken to be diagonal with common standard deviation 0.1%. In each case, prior to performing the update, the prediction given all previous information is recorded, for purposes of ('out-of-sample') comparison with the actual realised values.

The result is a prediction of daily return y_i, where it may be shown that investment (positive or negative) in proportion to the forecast $\hat{y}_i$ is optimal.

The cumulative returns (on the logarithmic scale) for the resulting filtered trading rule are summarised in the following graph, with the raw ISEQ itself over the same period included for comparison.

The risk-adjusted Sharpe ('Reward to Variability') ratio is computed to be approximately 100%, nearly twice the value attained by mere long positions on most major world indices during a typical 'Bull run' period. While this figure is not perhaps has high as has been reported in relation to some other strongly performing technical rules which have been suggested in the literature, given the simplicity of the approach, the results here would appear to be promising.

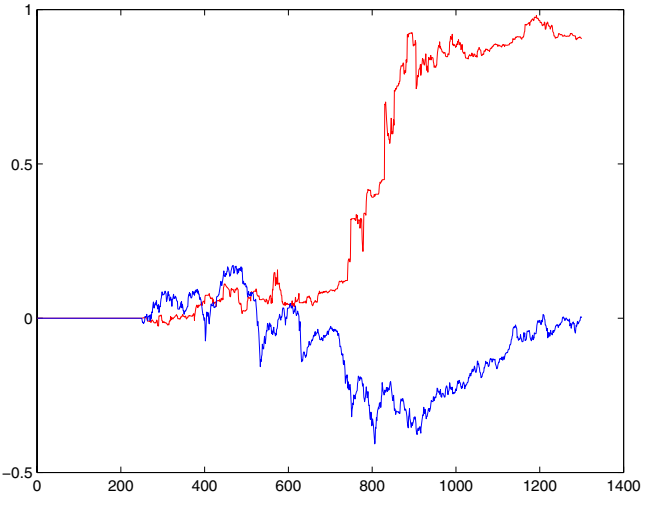


Fig. 1. Cumulative Returns for ISEQ and RBF-Kalman fund, 2001-5

4 Discussion and Conclusions

The primary contribution here is the suggestion of a simple paradigm which combines the power of Neural Network modeling (specifically, RBF networks with fixed centers) with the effectiveness of Kalman Filtering for tracking time-varying unobservable systems.

The main user-specified parameters required for application of the method are the grid spacing, the RBF bandwidth, and the 'signal-to-noise' ratio of the system. These should be optimised during a 'pre-online' phase, here taken to be a timespan of one year. While here this step has been carried out via trial-and-error on a small number of combinations only, it is believed that some form of Evolutionary Algorithm might be expected to greatly improve on this. Also worth noting is the special case of a static system, or 'signal-to-noise' ratio zero, which (while not being presented here) was found to be wholly inadequate for investment purposes.

Also worth noting is that as favourable as the performance presented in the previous section appears, visual inspection of the graph of returns suggests some degree of serial correlation in performance, which might suggest scope for further improvement. On the other hand, if the short period high growth on the graph is omitted from the sample, the value of the Sharpe Ratio increases significantly, sugesting a type of performance characterised by generally stable growth punctuated by a few sharp rises. [It is felt that few investors would complain about this!]

On the whole, it is felt that the preliminary findings reported here appear to suggest that this simple approach merits further investigation.

References

1. Edelman,D. and Davy,P. (2004). "Adaptive Technical Analysis in the Financial Markets using Machine Learning: a Statistical View" *Applied Intelligent Systems.* Springer series: Studies in Fuzziness and Applied Computing, pp. 1-16
2. Haykin,S (2001). *Kalman Filters and Neural Networks* New York: Wiley
3. LeBaron,B. et al (1992) "Simple Technical Trading Rules and the Stochastic Properties of Stock Returns", *Journal of Finance* vol. 47, no. 5 pp 1731-1764
4. Lo,A. et. al (2000). "Foundations of Technical Analysis: Computational Algorithms, Statistical Inference, and Empirical Implementation" *Journal of Finance* vol 55 no. 4 pp 1705-1765
5. Simon, H.A. (1990) A mechanism for social selection and successful altruism, *Science* 250 (4988): 1665-1668.

Business Intelligence for Strategic Marketing: Predictive Modelling of Customer Behaviour Using Fuzzy Logic and Evolutionary Algorithms

Andrea G.B. Tettamanzi[1], Maria Carlesi[2], Lucia Pannese[2], and Mauro Santalmasi[2]

[1] Università degli Studi di Milano
Dipartimento di Tecnologie dell'Informazione
via Bramante 65, I-26013 Crema, Italy
andrea.tettamanzi@unimi.it
[2] imaginary s.r.l.
c/o Acceleratore d'Impresa del Politecnico di Milano,
Via Garofalo 39, I-20133 Milan, Italy
info@i-maginary.it

Abstract. This paper describes an application of evolutionary algorithms to the predictive modelling of customer behaviour in a business environment. Predictive models are represented as fuzzy rule bases, which allows for intuitive human interpretability of the results obtained, while providing satisfactory accuracy. An empirical case study is presented to show the effectiveness of the approach.

Keywords: Business Intelligence, Data Mining, Modelling, Strategic Marketing, Forecast, Evolutionary Algorithms.

1 Introduction

Companies face everyday problems related to uncertainty in organizational planning activities: accurate and timely knowledge means improved business performance. In this framework, business intelligence applications represent instruments for improving the decision making process within the company, by achieving a deeper understanding of market dynamics and customers' behaviour.

Particularly, in the fields of business and finance, the executives can improve their insight of market scenarios by foreseeing customers' behaviour. This information allows to maximize revenues and manage costs through an increase in the effectiveness and efficiency of all the strategies and processes which involve the customers.

Predictions about customers' intentions to purchase a product, about their loyalty rating, the gross operating margins or turnover they will generate, are fundamental for two reasons. Firstly, they are instrumental to an effective planning of production volumes and specific promotional activities. Secondly, the comparison of projections to actual results allows to spot meaningful indicators, useful for improving performance.

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 233–240, 2007.
© Springer-Verlag Berlin Heidelberg 2007

A versatile solution for business intelligence, called iCLIP (imaginary's Client Profiler), has been developed by imaginary, a Milan-based company specialising in knowledge management. This paper describes the fuzzy-evolutionary approach to data mining used by iCLIP, in particular the MOLE engine, and presents the results of a case study on customer turnover prediction.

2 The Context

Traditional methods of customer analysis, like segmentation and market research, provide static knowledge about customers, which may become unreliable in time. A competitive advantage can be gained by adopting a data-mining approach whereby predictive models of customer behaviour are learned from historical data. Such knowledge is more fine-grained, in that it allows to reason about an individual customer, not a segment; furthermore, by re-running the learning algorithm as newer data become available, such an approach may be made to take a continuous picture of the current situation, thus providing dynamic knowledge about customers.

iCLIP uses evolutionary algorithms (EAs) for model learning, and expresses models as fuzzy rule bases. EAs are known to be well-suited to tracking optima in dynamic optimization problems [5]. Fuzzy rule bases have the desirable characteristic of being intelligible, as they are expressed in a language typically used by human experts to express their knowledge.

3 The Approach

In the area of business intelligence, data mining is a process aimed at discovering meaningful correlations, patterns, and trends between large amounts of data collected in a dataset. Once an objective of strategic marketing has been established, the system needs a wide dataset including as many data as possible not only to describe customers, but also to characterize their behaviour and tracing their actions.

The model is determined by observing past behaviour of customers and extracting the relevant variables and correlations between data and rating (dependent variable) and it provides the company with projections based on the characteristics of each customer: a good knowledge of customers is the key for a successful marketing strategy.

The tool is based on the use of EAs which recognise patterns within the dataset, by learning classifiers represented by sets of fuzzy rules.

3.1 Fuzzy Classifiers

Each classifier is described through a set of fuzzy rules. A rule is made by one or more antecedent clauses ("IF ...") and a consequent clause ("THEN ..."). Clauses are represented by a pair of indices referring respectively to a variable and to one of its fuzzy sub-domains, i.e., a membership function.

Using fuzzy rules makes it possible to get homogenous predictions for different clusters without imposing a traditional partition based on crisp thresholds, that often do not fit the data, particularly in business applications. Fuzzy decision rules are useful in approximating non-linear functions because they have a good interpolative power and are intuitive and easily intelligible at the same time. Their characteristics allow the model to give an effective representation of the reality and simultaneously avoid the "black-box" effect of, e.g., neural networks.

The output of the iCLIP application is a set of rules written in plain consequential sentences. The intelligibility of the model and the high explanatory power of the obtained rules are useful for the firm, in fact the rules are easy to be interpreted and explained, so that an expert of the firm can clearly read and understand them. An easy understanding of a forecasting method is a fundamental characteristic, since otherwise the managers are reluctant to use forecasts [1]. Moreover, the proposed approach provides the managers with an information that is more transparent for the stakeholders and can easily be shared with them.

3.2 The Evolutionary Algorithm

EAs are a broad class of stochastic optimization algorithms, inspired by biology and in particular by those biological processes that allow populations of organisms to adapt to their surrounding environment: genetic inheritance and survival of the fittest. Recent texts of reference and synthesis in the field of EAs are [7,6,3,2].

The iCLIP system incorporates an EA for the design and optimization of fuzzy rulebases that was originally developed to automatically learn fuzzy controllers [9,8], then was adapted for data mining [4] and is at the basis of MOLE, a general-purpose distributed engine for modelling and data mining based on EAs and fuzzy logic.

A MOLE classifier is a rule base, of up to 256 rules, each comprising up to four antecedent and one consequent clause. Up to 256 input variables and one output variable can be handled, described by up to 16 distinct membership functions each. Membership functions for input variables are trapezoidal, while membership functions for the output variables are triangular.

An island-based distributed EA is used to evolve classifiers. The sequential algorithm executed on every island is a standard generational replacement, elitist EA. Crossover and mutation are never applied to the best individual in the population.

Genetic Operators. The recombination operator is designed to preserve the syntactic legality of classifiers. A new classifier is obtained by combining the pieces of two parent classifiers. Each rule of the offspring classifier can be inherited from one of the parent programs with probability 1/2. When inherited, a rule takes with it to the offspring classifier all the referred domains with their membership functions. Other domains can be inherited from the parents, even if they are not used in the rule set of the child classifier, to increase the size of the offspring so that their size is roughly the average of its parents' sizes.

Like recombination, mutation produces only legal models, by applying small changes to the various syntactic parts of a fuzzy rulebase.

Migration is responsible for the diffusion of genetic material between populations residing on different islands. At each generation, with a small probability (the migration rate), a copy of the best individual of an island is sent to all connected islands and as many of the worst individuals as the number of connected islands are replaced with an equal number of immigrants.

3.3 Fitness

Modelling can be thought of as an optimization problem, where we wish to find the model M^* which maximizes some criterion which measure its accuracy in predicting $y_i = x_{im}$ for all records $i = 1, \ldots, N$ in the training dataset. The most natural criteria for measuring model accuracy are the mean absolute error and the mean square error.

One big problem with using such criteria is that the dataset must be *balanced*, i.e., an equal number of representative for each possible value of the predictive attribute y_i must be present, otherwise the underrepresented classes will end up being modeled with lesser accuracy. In other words, the optimal model would be very good at predicting representatives of highly represented classes, and quite poor at predicting individuals from other classes.

To solve this problem, MOLE divides the range $[y_{\min}, y_{\max}]$ of the predictive variable into 256 bins. The bth bin, X_b, contains all the indices i such that

$$1 + \lfloor 255 \frac{y_i - y_{\min}}{y_{\max} - y_{\min}} \rfloor = b. \tag{1}$$

For each bin $b = 1, \ldots, 256$, it computes the mean absolute error for that bin

$$\mathrm{err}_b(M) = \frac{1}{\|X_b\|} \sum_{i \in X_b} |y_i - M(x_{i1}, \ldots, x_{i,m-1})|, \tag{2}$$

then the total absolute error as an integral of the histogram of the absolute errors for all the bins, $\mathrm{tae}(M) = \sum_{b:\|X_b\| \neq 0} \mathrm{err}_b(M)$. Now, the mean absolute error for every bin in the above summation counts just the same no matter how many records in the dataset belong to that bin. In other words, the level of representation of each bin (which, roughly speaking, corresponds to a class) has been factored out by the calculation of $\mathrm{err}_b(M)$. What we want from a model is that it is accurate in predicting all classes, independently of their cardinality.

The fitness used by the EA is given by $f(M) = \frac{1}{\mathrm{tae}(M)+1}$, in such a way that a greater fitness corresponds to a more accurate model.

3.4 Selection and Overfitting Control

In order to avoid overfitting, the following mechanism is applied: the dataset is split into two subsets, namely the training set and the test set. The training set

is used to compute the fitness considered by selection, whereas the test set is used to compute a test fitness. Now, for each island, the best model so far, M^*, is stored aside; at every generation, the best individual with respect to fitness is obtained,

$$M_{\text{best}} = \text{argmax}_i f(M_i).$$

The test fitness of M_{best}, $f_{\text{test}}(M_{\text{best}})$, is computed and, together with $f(M_{\text{best}})$, it is used to determine an optimistic and a pessimistic estimate of the real quality of a model: for all model M, $f_{\text{opt}}(M) = \max\{f(M), f_{\text{test}}(M)\}$, and $f_{\text{pess}}(M) = \min\{f(M), f_{\text{test}}(M)\}$. Now, M_{best} replaces M^* if and only if $f_{\text{pess}}(M_{\text{best}}) > f_{\text{pess}}(M^*)$, or, in case $f_{\text{pess}}(M_{\text{best}}) = f_{\text{pess}}(M^*)$, if $f_{\text{opt}}(M_{\text{best}}) > f_{\text{opt}}(M^*)$.

Elitist linear ranking selection, with an adjustable selection pressure, is responsible for improvements from one generation to the next. Overall, the algorithm is elitist, in the sense that the best individual in the population is always passed on unchanged to the next generation, without undergoing crossover or mutation.

4 A Case Study on Customer Turnover Modelling

The iCLIP system has been applied to the predictive modelling of the turnover generated by customers of an Italian medium-sized manufacturing corporation operating in the field of wood and its end products. A pilot test was performed to demonstrate the feasibility of an innovative approach to customers modelling in turnover segments. In order to reduce time and costs, the traditional statistical analysis of data was skipped. Classifying customers into turnover segments can be useful not only to plan the activities and be aware of the returns for the next period, but also to identify characteristics which describe different patterns of customers, to recognise strategic and occasional customers, to target commercial/marketing activities and so on.

iCLIP was used to develop a predictive model foreseeing customers' turnover segments for a quarter using historical data of the year before the analysis. Customers were classified into three quarterly turnover segments:

1st segment: turnover >50,000 euro/quarter;
2nd segment: turnover between 10,000 and 50,000 euro/quarter;
3rd segment: turnover <10,000 euro/quarter.

Historical data on turnover generated by each customer c were available in the form of monthly turnovers m_{ip}^c, for the ith month ($i = 1, \ldots, 24$) and four homogeneous classes of products, $p = 1, \ldots, 4$. These data have been aggregated on a quarterly basis, giving a vector of quarterly turnovers q_{jp}^c, to be used in order to perform an analysis of the 12-months trend-cycle. Data were adjusted seasonally, since the observations relating to the month of August, when most businesses shut down for vacations, were supposed to be not significant.

The dataset given to MOLE as input is extracted from the de-seasonalised quarterly historical data q_{jp}^c as follows: a sliding window of one year (i.e., four

quarters) plus the turnover segment y_{j+4} for the forward quarter (based on total customer turnover) is used to extract a staggered set of records in the form

$$q^c_{j1}, \ldots, q^c_{j4}, \ldots, q^c_{j+3,1}, \ldots, q^c_{j+3,4}, y_{j+4}, \tag{3}$$

for $j = 1, \ldots, 19$. Such a record provides a summary of a year of activity by a customer, along with the associate value of the predictive variable y_{j+4}.

With reference to the three selected turnover segments for this pilot test, MOLE was also given a customer segment, calculated by aggregating partial turnovers related to every single product during the following period. For example, if we consider quarterly data for Q1, Q2, Q3, Q4, then the turnover segment is calculated based on the sum of four single products turnovers in the first quarter of the following year.

Finally, data concerning customer's industrial sector, geographical location, and average quarterly turnover during the previous year were also added to each record.

The resulting dataset, consisting of 19 distinct records for each customer, i.e., 68,666 records overall, was fed to the MOLE evolutionary engine for model learning, with 4 islands of 100 individuals each connected according to a ring topology and a migration rate of 0.1; for each island, the mutation rate was set to 0.1 and the crossover rate to 0.5.

4.1 Discussion of the Results

In order to simplify the procedures for this preliminary test, the authors assigned meaningful labels (e.g., low, medium, high, etc.) to the membership functions generated by MOLE, although such labels should normally be established jointly with the customer.

The y_{j+4} variable (the "rating") represents the turnover segment for the following quarter.

The algorithm underlying the model evaluates all the rules at the same time and provides forecasts by calculating the average between values assigned to the rating in the consequent of every rule, weighted by the degree of satisfaction of the antecedent.

Model output is usually generated as result of the interaction between more than one fuzzy rule weighted using the corresponding satisfaction degree.

Following some of the rules representing the selected model are proposed:

IF $q_{j+1,3}$ **is** medium **AND** $q_{j+2,3}$ **is** very high
THEN y_{j+4} **is** first segment

IF $q_{j+3,3}$ **is** medium-low **AND** $q_{j+1,4}$ **is** very high
THEN y_{j+4} **is** first segment

IF $q_{j+1,3}$ **is** very low **AND**
$q_{j+2,2}$ **is** medium-high **AND**
$q_{j+3,3}$ **is** medium-high **AND**
$q_{j+1,1}$ **is** any
THEN y_{j+4} **is** first segment

The managers of the company could easily evaluate the correlations suggested by the rules. For example, analysing the presented rules, it is possible to recognize the trend of purchasing for product 3: purchases of this product have a high frequency and the customer which has given to the company a turnover even if medium in a recent period probably will generate an high turnover in the next period.

4.2 Validation of the Model

The model thus determined using data up to September 2005 has been employed to predict future turnover for every customer in the fourth quarter 2005; then, at the beginning of 2006, the predictions have been compared with actual data referring to the same period: the model correctly hits 2,672 records out of 3,614 and the fitness of the model to the data is 0.44, while the total error is 1.27.

The error has been compared to the one obtained using simple forecasts normally used by the company and based exclusively on the average turnover during the previous year.

The error of iCLIP in foreseeing turnover resulted significantly lower for segments 1 and 2, which are the most strategic for the company, as they comprise the customers that make up for the greatest part (almost 70%) of the total turnover. Table 1 shows the distribution of the error with iCLIP (concentrated in the third segment) and the difference between the results obtained using iCLIP versus the estimation based on the average turnover during the previous year.

Table 1. Comparison between error in predictions obtained on the basis of average turnover during the previous year on one hand and using iCLIP on the other hand

Segment	Error using Average	Error using iCLIP
1 (>50,000 EUR/Q)	0.71	0.5
2 (10,000–50,000 EUR/Q)	0.54	0.5
3 (<10,000 EUR/Q)	0.03	0.26

5 Conclusions

The pilot test was implemented without a preliminary elaboration of the data. The positive results allow the authors to expect further improvements through an analysis of the data, to identify other potential explicative variables or useful information about customer behaviour, i.e., for example, frequency of purchases, etc.

iCLIP forecasts provide the company with new qualitative knowledge of customers, also due to the flexibility of the instrument that can be used in many different applications. Moreover, these additional information are always up to date, since the system automatically considers new data as soon as they are added to the dataset.

Dynamic predictive knowledge of customers is the most important factor to develop an effective marketing strategy. In fact, the aim of business intelligence is improving the comprehension of market dynamics and actors, and these forecasts allow the firm to understand the development of customer preferences and complete the framework of internal knowledge of the company in order to be able to answer the continuous new customer needs. The company has thus the opportunity to effectively plan production volumes and financial flows for the following period and can target specific promotional and cross-selling activities. Moreover, the executives have an instrument to evaluate the policies adopted by sellers in different regions: the gap between forecasts and real data can be the term of comparison between the performance of different sellers.

The decision-making process within the company takes an important advantage in terms of invested time and resources, and finally the effectiveness of marketing activities can considerably increase.

References

1. J. S. Armstrong, R. J. Brodie, and S. H. McIntyre. Forecasting methods for marketing: Review of empirical research. *International Journal of Forecasting*, 3:335–376, 1987.
2. T. Bäck. *Evolutionary algorithms in theory and practice*. Oxford University Press, Oxford, 1996.
3. Thomas Bäck, David Fogel, and Zbigniew Michalewicz. *Evolutionary Computation*. IoP Publishing, Bristol, 2000.
4. M. Beretta and A. Tettamanzi. Learning fuzzy classifiers with evolutionary algorithms. In G. Pasi A. Bonarini, F. Masulli, editor, *Soft Computing Applications*, pages 1–10. Physica Verlag, Heidelberg, 2003.
5. Jürgen Branke. *Evolutionary Optimization in Dynamic Environments*. Kluwer Academic Publishers, Dordrecht, 2001.
6. Kenneth A. DeJong. *Evolutionary Computation: A unified approach*. MIT Press, Cambridge, MA, 2002.
7. Agoston E. Eiben and J. E. Smith. *Introduction to Evolutionary Computing*. Springer-Verlag, Berlin, 2003.
8. A. Tettamanzi R. Poluzzi, G. G. Rizzotto. An evolutionary algorithm for fuzzy controller synthesis and optimization based on sgs-thomson's W.A.R.P. fuzzy processor. In L. A. Zadeh E. Sanchez, T. Shibata, editor, *Genetic algorithms and fuzzy logic systems: Soft computing perspectives*. World Scientific, Singapore, 1996.
9. A. Tettamanzi. An evolutionary algorithm for fuzzy controller synthesis and optimization. In *IEEE International Conference on Systems, Man and Cybernetics*, volume 5/5, pages 4021–4026. IEEE Systems, Man, and Cybernetics Society, 1995.

Particle Swarm Optimization for Object Detection and Segmentation

Stefano Cagnoni, Monica Mordonini, and Jonathan Sartori

Università di Parma
Dipartimento di Ingegneria dell'Informazione
viale G.Usberti 181/A, 43100 Parma - Italy
{cagnoni,monica}@ce.unipr.it, jsartori@libero.it

Abstract. In this paper we describe results of a modified Particle Swarm Optimization (PSO) algorithm which has been applied to two image analysis tasks. In the former, accurate region-based segmentation is obtained by analyzing the cumulative results of several runs of the algorithm. In the latter, the fast-convergence properties of the algorithm are used to accurately locate and track an object of interest in real time.

1 Introduction

Since its introduction in the late 90's [1,2], Particle Swarm Optimization (PSO) has increasingly attracted researchers for its efficiency in locating maxima of even highly multi-modal functions.

While the basic PSO algorithm aims at finding a single optimum point within the fitness landscape under exploration, several applications require that more than one optimum be found or that the swarm spread over a whole area of interest, featuring high fitness values, as uniformly as possible. This has led to the definition of several variants of basic PSO, in which particles are subdivided into a pre-defined number of subswarms, based on some clustering technique [3,4,5], or through speciation [6,7,8,9], to achieve a dynamical reconfiguration of the swarm and allow for an arbitrary number of regions of interest within the search space.

This situation is typical of object recognition tasks, where the goal is to identify all possible occurrences, within an image, of an object of interest which is characterized by a set of specific, even if generally fuzzily defined, features. Similarly, region-based segmentation requires that several regions with homogeneous features be accurately located.

In this paper we describe two image analysis applications which are solved by methods based on PSO variants adapted to the specific requirements of the problems under consideration.

2 PSO for Object Detection and Segmentation

The two tasks which were used to evaluate the potential of PSO for application to image analysis are two typical tasks in computer vision: region-based segmentation and object detection.

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 241–250, 2007.
© Springer-Verlag Berlin Heidelberg 2007

The first task is the 'pasta segmentation problem', proposed as subject of a competition at GECCO 2006[1]. In such a competition, each image in a set of 12, obtained in different lighting conditions and presenting lighting artifacts, as the bright spot in Figure 2 (top left), were to be analyzed. In particular, the larger pieces of pasta, laid on complex backgrounds and mixed with smaller ones (pasta noise), were to be segmented using a binary pixel-classification strategy. The final classification was to be obtained by thresholding the image resulting from some pre-processing of the original one (see Figure 1).

The second task on which PSO was tested was license-plate detection. The goal here was to locate the license plate within rear views of cars acquired under different lighting conditions and camera positions. The same dataset which had been used in developing the APACHE license-plate recognition system [10] was used as benchmark, in order for results to be compared with those obtained by the plate detection stage of such a system. Additionally, further experiments on video sequences were also made, in which the goal was tracking the license plate through the frames in real time, once it had been located.

Even if the two tasks are semantically different, they share some common lower-level features, which allowed us to apply very similar versions of PSO to solve the two problems. In particular, in both cases the basic step requires that the image be explored, to focus on regions where interesting features (i.e., features which are expected to characterize the objects we want to locate or segment) can be detected. The main goal of our work was therefore to evaluate to which extent the efficiency of PSO-based search could be exploited within the two applications.

The following subsection will describe how the basic PSO equations have been modified to fit the requirements of our applications and to implement the basic step, common to both applications, before describing the applications in details in the following two sections.

2.1 PSO Fitness and Velocity-Update Equations

In the basic PSO algorithm, the fitness function is punctually coincident with the function which is to be optimized. In analyzing images using PSO, the search space being the image, using such a local fitness function would lead to explorations which would be extremely sensitive to noise and possibly misleading. If fitness evaluation were just pixel-based, a meaningless isolated pixel yielding high fitness as a result of noise could attract and trap the whole swarm into its neighborhood.

In the applications under consideration, PSO is required to produce a uniform distribution of particles over each region of interest. To induce such a behavior, we have modified the basic PSO algorithm in two directions:

- forcing division of the swarm into sub-swarms (defined as subsets of the swarm, within which the distance between any particle and the closest one is below a preset threshold; sub-swarms change dynamically as particles move), able to converge towards different regions of interest;
- favoring dispersion of the particles all over the regions of interest.

[1] See `http://cswww.essex.ac.uk/staff/rpoli/GECCO2006/`

Using the so-called *K-means PSO* [5], in which clusters of particles are formed based on their proximity within the search space, allowed us to achieve the former goal. Achieving the latter required that both the fitness function and the velocity-update equation be modified.

As concerns the fitness function to be maximized, we have added a $local$ fitness term, which evaluates how "interesting" the neighborhood of one pixel is, to the traditional $punctual$ fitness function, whose value is computed based only on information carried by the pixel under consideration:

$$fitness(x,y) \;=\; punctual_fitness(x,y) \;+\; local_fitness(x,y) \tag{1}$$

The $local_fitness$ term depends on the number of particles in the sub-swarm, with high punctual fitness, which are near the pixel located in (x,y) and is given by:

$$local_fitness \;=\; K_0 \,*\, number_of_neighbors \tag{2}$$

where $number_of_neighbors$ is the number of particles within a pre-defined neighborhood of (x,y) and K_0 is a constant.

This way, the particles are attracted towards the areas where a larger amount of pixels meet the punctual requirement, keeping away from isolated noisy pixels. This modification enhances the density of particles in the most interesting regions. To cover the whole extension of these regions and not only small areas within them, we needed to modify also the basic PSO velocity-update equation from:

$$\begin{aligned}\mathbf{vP}(t) = w * \mathbf{vP}(t-1) + \\ C_1 * rand() * [\mathbf{X}_{best} - \mathbf{X}(t-1)] \;+ \\ C_2 * rand() * [\mathbf{X}_{gbest} - \mathbf{X}(t-1)]\end{aligned} \tag{3}$$

where $\mathbf{vP}$ is the velocity of the particle, C_1, C_2 are two positive constants, w is the inertia weight, $\mathbf{X}$ is the position of the particle, $\mathbf{X}_{best}$ is the best-fitness position reached by the particle up to time $t-1$, $\mathbf{X}_{gbest}$ is the best-fitness point ever found by the whole swarm, to:

$$\mathbf{vP}^*(t) \;=\; \mathbf{vP}(t) \;+\; \mathbf{repulsionP} \tag{4}$$

The $repulsion$ term is computed, separately along each axis, as

$$|\mathbf{repulsion}(i,j)| \;=\; REPULSION_RANGE \;-\; |\mathbf{X}_i - \mathbf{X}_j| \tag{5}$$

where i and j are the particle indices and $REPULSION_RANGE$ is the maximum distance within which the particles may interact. Values of **repulsion**(i,j) are set to 0 for distances between i and j larger than $REPULSION_RANGE$.

The global repulsion term $\mathbf{repulsionP}$ for particle P is the average of all repulsion terms deriving from the presence of other particles in its neighborhood.

$$\mathbf{repulsionP} \;=\; \frac{1}{n} \sum_{j=1}^{N} \mathbf{repulsion}(\mathbf{P},\mathbf{j}) \tag{6}$$

N being the number of particles in the swarm and n being the number of particles within the neighborhood of P defined by $REPULSION_RANGE$.

Finally, one last change has been made to the standard PSO algorithm, aimed at producing more stable sub-swarms: the possibility for a particle with high punctual and local fitness to stand still. In other words, if a particle with a high punctual fitness lies within a region with a high density of particles, then it has a probability of standing still, which is linearly dependent on such a density.

Such a probability is estimated as:

$$P\{vP(t) = 0\} = \frac{n}{N} \tag{7}$$

3 PSO-Based Image Segmentation

As described in the previous section, the $punctual_fitness$ is the fitness which could be attributed to a pixel (a location in the search space), based only on its intrinsic properties. In the pasta segmentation problem, this translates into a function which measures the similarity of the pixel color to the expected color of pasta pieces or, better, its belonging to a three-dimensional region in the RGB space centered around such a color prototype, which is expressed as:

$$\begin{aligned} &if \quad (|r(x,y) - g(x,y)| < 30 \quad and \quad r(x,y) - b(x,y) \; > \; 60) \\ &\qquad punctual_fitness = 30 - |r(x,y) - g(x,y)| \\ &else \\ &\qquad punctual_fitness = 0 \end{aligned}$$

where $r(x,y)$, $g(x,y)$ and $b(x,y)$ are the red, green and blue values, respectively, of the pixel located in (x,y).

Since the aim of the application was to obtain an accurate segmentation, up to pixel precision, and given the rather large size of the input images and the consequent large number of pixels belonging to the objects of interest, it is obvious that PSO could not produce the final solution directly. Instead, it was used in the pre-processing stages preceding the final thresholding stage which produces the actual output.

Following the PSO rules modified as previously described, the particles will tend to move towards larger pasta regions and to stay around there. If one performs a number of runs of a PSO algorithm, assigning each pixel a score which is directly proportional to the number of times a particle walks through (or stays on, if the particle stands still) it, the probability of belonging to a large pasta piece can be estimated for each pixel. To better estimate such a probability, and to avoid possible polarizations of results due to the choice of the initial particle locations, each run should start with a different random initialization of the whole swarm.

To give globally higher importance to the regions with higher density of pasta pixels, and to regularize results, we decided to extend the 'influence area' of each particle from just the current pixel to a larger neighborhood. In other words, when a particle visits a pixel the score of the current pixel is increased, as well as the score of its neighbors, by a lower amount roughly proportional to their distance from the current pixel. Finally,

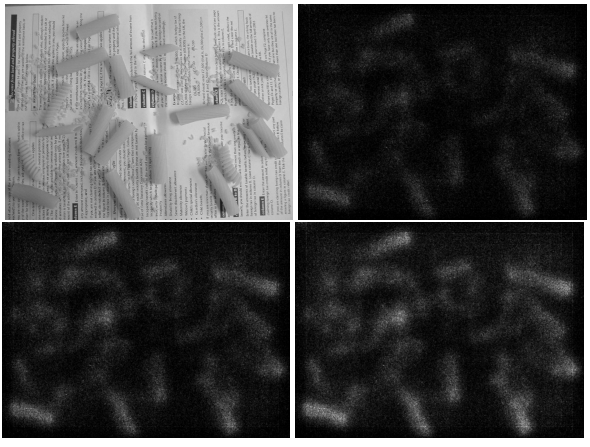


Fig. 1. Original image (top left) and results of global search after 500 runs (top right), 750 runs (bottom left), and 1000 runs (bottom right)

we stretch the score distribution within the image by further awarding the pixels whose score is above a threshold by multiplying it by a fixed factor $F > 1$, whilst the scores which are below the threshold are reduced by multiplying them by a different factor $G < 1$.

In Figure 1, the score associated with each pixel is represented as a grey-scale image. Areas which eventually end up by having high density of light pixels (i.e., high scores) correspond to pasta regions. The final result of this stage, that we termed *global search*, is shown in Figure 2.

This way, the areas where large pieces of pasta are most likely to be found, have been grossly detected on the whole image; it is now necessary to focus the attention on such areas to achieve a final refinement of their segmentation. To do so, an algorithm which is very similar to the one used in the previous stage is applied; this time the domain where the swarm can move is limited to smaller regions surrounding pixel clusters whose score was above the threshold in the last phase of the global search stage.

These are rectangular regions, extracted as follows:

- a neighborhood with high density of significant pixels is detected;
- the neighborhood is extended until a bounding box is found for the relevant pixel cluster;
- the bounding box is extended by $1/3$ in each direction, to cope with possible false negatives close to the boundary of the piece of pasta corresponding to the cluster.

In initializing this stage, scores assigned to each pixels at the end of the global search are preserved. The result, obtained after running this local search on all significant regions, is shown in Figure 2, along with the results of the final segmentation, obtained by thresholding the results of the local search.

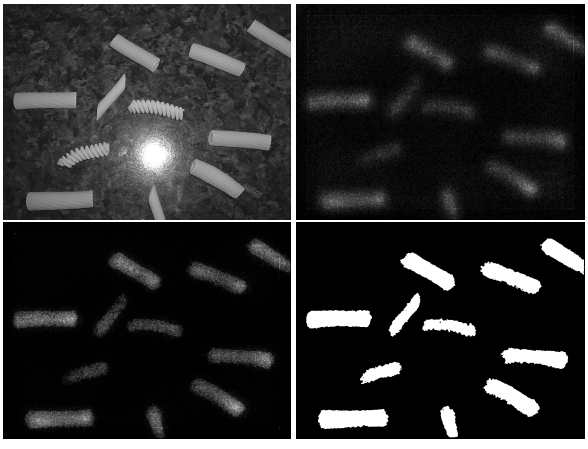


Fig. 2. Original image (top left), results of global search (top right), results of local search (bottom left), and final segmentation (bottom right)

4 PSO-Based Pattern Localization

In the license plate detection problem, the low-level feature on which detection can be based is the density of high-level values of the horizontal gradient, which correspond to the frequent discontinuities between high- and low-intensity pixels (or vice-versa), due to the presence, in the plate, of symbols or symbol elements, which can be encountered when the image is scanned row-wise. Since a color image is available, we can use both color and gradient information, by firstly considering only those pixels which satisfy the typical features of plates (black characters on a white background for the most recent European-standard plates), and then considering gradient information.

Therefore, the punctual fitness of a pixel, on which PSO-based plate search will rely has been defined in this application as:

$$
\begin{aligned}
&if \quad (|r(x,y) - g(x,y)| > 30 \\
&\quad or\ |r(x,y) - b(x,y)| > 30 \\
&\quad or\ |g(x,y) - b(x,y)| > 30) \\
&\qquad punctual_fitness = 0; \\
&else\ \{\ right_gradient = |intensity(x,y) - intensity(x+1,y)|; \\
&\qquad left_gradient = |intensity(x,y) - intensity(x-1,y)|; \\
&\qquad if\ (\ right_gradient > left_gradient\ ) \\
&\qquad\quad punctual_fitness = right_gradient; \\
&\qquad else\ punctual_fitness = left_gradient; \}
\end{aligned}
$$

The basic PSO step used to solve this problem was virtually the same as in the application described previously. However, it was used within a different algorithm, which is, as well, divided into a global and a local exploration stage in which, after the



Fig. 3. License plate detection. Original image (top left), the sub-swarms at the end of the global search super-imposed on the gradient image (top right), the swarm at the end of the local search super-imposed on the gradient image (bottom left), and the bounding box corresponding to the license plate super-imposed on the input image (bottom right).

most promising areas are firstly located, the exploration of those region is then refined to determine whether they actually represent a plate.

In the global search stage, we let the swarm fly over the image until at least one sub-swarm of size greater than a prefixed threshold (50% of the number of particles in the whole swarm) has formed, or a given number of iterations has been reached.

Then, in the subsequent stage, a local search is performed in the areas where sub-swarms of sufficient dimension have formed (at least 3 particles), starting from the region occupied by the most numerous one; during this second stage, we (i) restrict the search to the bounding boxes enclosing the sub-swarms, defined as in the previous application, by clipping particle positions at the boundaries of the bounding box, (ii) re-initialize the search activating a new (full-size) swarm, and (iii) run the search for a pre-set number of iterations. Also in this case, we refer to this second stage as the *local search* stage.

At the end of the local search, a new bounding box, containing all the particles having high fitness, is defined. If this box has a width:height ratio close to 5:1 (the ratio which is typical of a license plate), then the plate is considered to have been found. Otherwise, the swarm is expanded along its two dimensions, by forcing low-fitness particles to move only horizontally or vertically, in order to reach higher-fitness points and, possibly, to let the bounding box reach the expected aspect ratio; in case of failure the current region is discarded and the next area detected during the global search is explored.

Figure 3 shows the original image, along with the results of the global and local search, and the final result of the PSO-based algorithm.

5 Experimental Results

The two applications we have considered were aimed at evaluating the performances which could be obtained using PSO as a search algorithm in image analysis tasks.

For both applications the parameters related with swarm motion and fitness equation were set as follows:

$$w = 0.8,\ \ C_1 = C_2 = 2.0,\ \ K_0 = 5.0,\ \ REPULSION_RANGE = 7 \qquad (8)$$

In the application to pasta segmentation $G = 1.0$ while $F = 2.0$ during global search and $F = 3.0$ during local search.

Considering the different goals of the two applications, it is quite clear that plate detection is a much less demanding application, in terms of computation time, with respect to pasta segmentation. Once the search has been successful, which most often happens in the first run, limited post-processing is required by the plate detection algorithm to define an accurate bounding box for the plate. On the contrary, pasta segmentation, which requires not only detection but also accurate segmentation of all objects of interest, requires many more runs of PSO to reach stable statistics on the number of visits to each pixel before thresholding can be applied. In fact, using PSO to perform virtually the whole segmentation task is a solution which seems to force PSO to perform beyond its nature of fast and effective search algorithm. This was clearly reflected by the computation time required by pasta segmentation, for which 25 seconds per image were needed on average on a 1.8 GHz Pentium4 PC having 1 GB of RAM. For every image, 1000 PSO runs, each lasting 500 generations (updates of the position of the whole swarm of 20 particles) were performed in the global search stage, while a number of runs proportional to the number of regions of interest extracted during the global search were performed during the local search.

Apart from computational inefficiency, segmentation was quite accurate, averaging an accuracy of 91.89% on the 12 images of the set under consideration, when the optimum threshold was chosen separately for each image, with values ranging from 86.65% to 97.71%. However, the robustness of the segmentation induced by the two PSO-based stages of the algorithm is such that accuracy was very little dependent on the threshold value, as shown by table 1.

In the license plate detection experiment, the data set on which tests were performed included 98 rear images of cars acquired with different backgrounds and lighting conditions. Given the stochastic nature of the algorithm, 10 runs of the algorithm were performed for each image in the data set, with swarms of 20 particles. The algorithm was able to detect the plate correctly (maximum distance of 3 pixels between the actual border of the license plate and the bounding box extracted by the algorithm) in 958 cases out of 980, with a success rate of 97.76%, much higher than the algorithm which was used in the APACHE system, whose success rate was below 90%.

A very interesting observation regarding the PSO-based solution with respect to a traditional computer vision algorithm relying on the same information is that, in the latter, the whole horizontal-gradient image has typically to be computed before the

Table 1. Average percent accuracy vs. threshold value for the pasta segmentation problem

Threshold	0.05	0.10	0.15	0.20	0.25	0.30	0.35	0.40	0.50	0.60	0.70	0.80	0.90
Accuracy	70.78	88.89	90.89	91.57	91.69	91.41	90.83	91.14	90.46	86.96	83.68	83.43	83.33

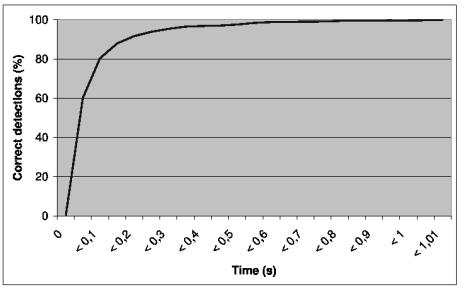


Fig. 4. Percentage of detections vs. time for the successful runs

analysis can start, while in the former the horizontal gradient is computed only for those pixels which were visited by the swarm, which, on average, means only as many times as 2% to 3% of the total number of pixels in the image.

Therefore, results in term of computation time were even more satisfactory. The average time required to detect a plate was 0.083 s for the successful runs (see Figure 4 for detailed statistics on the distribution of results in time).

The runs in which the plate was wrongly detected required on average 0.298 s, while the runs in which the plate could not be detected at all 1.521 s. The average time over the whole data set was 0.096 s per image, which means that, even without considering time correlation between images and performing the global search over the whole image for each frame, a video stream running at up to 10 frames per second could be analyzed on a 1.8 GHz Pentium4 PC having 1 GB of RAM.

To evaluate the real-time processing capabilities of the algorithm, in the presence of a tracking strategy, we made further tests on 7 video sequences recorded at 25 fps, of duration ranging from 1.5 to 5 s. To track the plate, after each successful detection, the swarm was initialized, in the subsequent frame, within a neighborhood of the region where the plate had been previously detected. If the search within the previous frame had been unsuccessful, initialization of the swarm could occur anywhere within the new frame. If plate search had been unsuccessful with such an initialization, a full search was performed all over the image. Also in this experiment each test was repeated 10 times for each sequence.

Table 2. Results of the PSO-based plate detection tracking algorithm

Sequence	N. of frames	Avg. time (s)	Success (%)
1	96	0.018	100
2	73	0.016	100
3	38	0.017	100
4	48	0.020	95.58
5	49	0.020	98.98
6	145	0.021	97.59
7	117	0.032	93.08
Total	566	0.022	97.83

The average processing time per frame was well below both limits of .04 s and .033 s required for real-time processing at 25 and 30 fps, respectively, even in the case of the most critical sequence, in which most failures occurred. Table 2 summarizes the results of this experiment.

6 Final Remarks

In this paper we have described a PSO-based approach to object detection and segmentation which can be considered rather general, as demonstrated by the fact that the two applications which have been described basically share the same algorithm, despite being semantically different. Of course, the fitness function has to be carefully defined, to reflect the peculiarities of the problem at hand. While the choice of the parameters of the PSO equations seems not to be critical (the same settings worked for both applications), problem-specific parameters (such as K_0) may depend on the the fitness function which is chosen and on how its output is scaled. The efficiency of PSO search is reflected by the real-time performances achieved in the object-recognition application.

References

1. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proc. IEEE Int. conf. on Neural Networks. Volume IV. (1995) 1942–1948
2. Shi, Y.H., Eberhart, R.: A modified particle swarm optimizer. In: Proc. IEEE Int. Conference on Evolutionary Computation. (1998) 69–73
3. Kennedy, J.: Stereotyping: Improving particle swarm performance with cluster analysis. In: Proc. IEEE Int. Conference on Evolutionary Computation. Volume II. (2000) 1507–1512
4. Veenhuis, C., Köppen, M.: Data swarm clustering. In Abraham, A., Grosan, C., Ramos, V., eds.: Swarm Intelligence in Data Mining. Volume 34 of Studies in Computational Intelligence. Springer (2006) 221–241
5. Passaro, A., Starita, A.: Clustering particles for multimodal function optimization. In: Proc. GSICE/WIVA. (2006) published on CD, ISSN 1970-5077.
6. Chow, C., Tsui, H.: Autonomous agent response learning by a multispecies particle swarm optimization. In: Proc. IEEE Congress on Evolutionary Computation. (2004) 778–785
7. Bird, S., Li, X.: Enhancing the robustness of a speciation-based PSO. In: Proc. IEEE Congress on Evolutionary Computation. (2006) 3185–3192
8. Yen, G., Daneshyari, M.: Diversity-based information exchange among multiple swarms in particle swarm optimization. In: Proc. IEEE Congress on Evolutionary Computation. (2006) 6150–6157
9. Leong, W., Yen, G.: Dynamic population size in PSO-based multiobjective optimization. In: Proc. IEEE Congress on Evolutionary Computation. (2006) 6182–6189
10. Adorni, G., Bergenti, F., Cagnoni, S., Mordonini, M.: License-plate recognition for restricted-access area control systems. In Foresti, G.L., Mähönen, P., Regazzoni, C.S., eds.: Multimedia Video-Based Surveillance Systems: Requirements, Issues and Solutions. Kluwer (2000) 260–271

Satellite Image Registration by Distributed Differential Evolution

I. De Falco[1], A. Della Cioppa[2], D. Maisto[1], U. Scafuri[1], and E. Tarantino[1]

[1] ICAR–CNR, Via P. Castellino 111, 80131 Naples, Italy
{ivanoe.defalco,ernesto.tarantino,umberto.scafuri}@na.icar.cnr.it
[2] DIIIE, University of Salerno, Via Ponte don Melillo 1, 84084 Fisciano (SA), Italy
adellacioppa@unisa.it

Abstract. In this paper a parallel software system based on Differential Evolution for the registration of images is designed, implemented and tested on a set of 2–D remotely sensed images on two problems, i.e. mosaicking and changes in time. Registration is carried out by finding the most suitable affine transformation in terms of maximization of the mutual information between the first image and the transformation of the second one, without any need for setting control points. A coarse–grained distributed version is implemented on a cluster of personal computers.

1 Introduction

Registration is a task in image processing used to match two or more pictures taken with different methods or at different times. Several techniques have been developed for various applications [1,2]. A highly interested field is remote sensing [3,4,5]. The goals of the paper are to design a software system for the registration of images, and to test it by means of a set of 2–D satellite images. Among the methods in literature, the use of an affine transformation [6] to "align" at best the two images appears of interest. Thus the problem is to find the best among all the possible transformations, each of which is represented by a set of real parameters. Evolutionary Algorithms (EAs) [7, 8] are a heuristic technique successfully used to face several multivariable optimization tasks and their use has been introduced in Image Registration as well, in particular in the remote sensing [9, 10, 11, 12]. In this paper we employ them to find the optimal combination of the parameter values involved in the affine transformation. Namely, a Differential Evolution (DE) [13] mechanism has been implemented. DE is a version of an EA which has proven fast and reliable in many applications [14].

There exist in literature several approaches based on either explicitly providing a set of control points [3,12,15] (including DE [16]) or in automatically extracting them from the image [17]. In contrast to those feature–based approaches, here we wish to examine DE ability to perform automatic image registration without making any use of control points.

Moreover we have designed and implemented a distributed scheme for DE based on the coarse–grained approach, which is often fruitful in EAs [20].

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 251–260, 2007.
© Springer-Verlag Berlin Heidelberg 2007

Paper structure is as follows: Section 2 describes the image registration problem and defines the affine transformation and the mutual information. Section 3 contains DE basic scheme and illustrates the application of our system to the image registration task. Section 4 depicts our distributed DE scheme, while Section 5 reports on the two remote sensing problems faced, i.e mosaicking and changes in time, and shows the results achieved by our distributed tool. Finally Section 6 contains conclusions and future works.

2 Image Registration

When registering remote sensing images two problems are typically faced, i.e. *Mosaicking* and *Change Discovery*. The former deals with spatially aligning two images of neighboring areas taken at the same time so as to obtain a larger image, whereas the latter consists in firstly aligning two images of about the same area but taken at different times, and then in pointing out the changes happened in that area within the difference timespan. In all cases, two choices must be made to carry out image registration. The first choice involves the kind of geometric transformation to be considered to find correlations between the given images, while the second one concerns the measure of match (MOM), i.e. the feature on the value of which the goodness of the registration is evaluated. Once made these choices, the MOM can be maximized by using suitable optimization algorithms.

Affine Transformation. The most frequently used transformation model in registration is the affine transformation. It is sufficiently general and can handle rotations, translations, scaling and shearing. It can be represented in the most general 3–D case as $\mathbf{x}' = A \cdot \mathbf{x} + \mathbf{b}$ where A is a 3×3 square matrix accounting for rotations, scalings and shears while $\mathbf{x}$, $\mathbf{x}'$ and $\mathbf{b}$ are 3–D arrays representing the original positions, the transformed ones and a translation vector.

Mutual Information. The most widely employed MOM is the Mutual Information (MI) [18,19], which represents the relative entropy of the two images to be registered. In general, given two random variables Y and Z, their MI is:

$$I(Y,Z) = \sum_{y,z} P_{Y,Z}(y,z) \cdot log \frac{P_{Y,Z}(y,z)}{P_Y(y) \cdot P_Z(z)} \tag{1}$$

where $P_Y(y)$ and $P_Z(z)$ are the marginal probability mass functions and $P_{Y,Z}(y,z)$ is the joint probability mass function. The MI registration criterion states that the image pair is geometrically aligned through a transformation $\mathbf{T}$ when $I(Z(\mathbf{x}), Y(\mathbf{T}(\mathbf{x})))$ is maximal. Thus, the aim is to maximize eq. (1).

3 Differential Evolution

Differential Evolution (DE) is a stochastic, population-based optimization algorithm [13]. Given a maximization problem with m real parameters, DE faces it by randomly initializing a population consisting of n individuals each made up by m real values. Then, the population is updated from a generation to the next

one by means of many different operators. Among them we have chosen the one referenced as *DE/rand/1/bin*. It takes into account the generic i–th individual in the current population, and randomly generates three integer numbers r_1, r_2 and r_3 in $[1, n]$ differing one another and different from i. Moreover, another integer number k in $[1, m]$ is randomly chosen. Then, starting from the i–th individual a new trial one i' is generated whose j–th component is given by:

$$x_{i',j} = x_{r_3,j} + F \cdot (x_{r_1,j} - x_{r_2,j}) \tag{2}$$

provided that either a random real number ρ in $[0.0, 1.0]$ is lower than a value CR (parameter of the algorithm, in the same range as ρ) or the position j under account is exactly k. If neither is verified then a copy takes place: $x_{i',j} = x_{i,j}$. F is a real and constant factor in $[0.0, 1.0]$ which controls the magnitude of the differential variation $(x_{r_1,j} - x_{r_2,j})$, and is a parameter of the algorithm.

The new individual i' is compared to the i–th in current population and the fittest is copied into the new population. This scheme is repeated for each individual and for a maximum number of generations g.

3.1 DE Applied to Image Registration

Encoding. We have decided to make use of the affine transformation model. Since the experiments reported in this paper make reference to couples of two–dimensional images, the problem consists in finding the most suitable best combination of six real–valued parameters. Therefore, any individual in the DE population is an array with six positions: $\mathbf{T} = (a_{11},\ a_{12},\ a_{21},\ a_{22},\ b_1,\ b_2)$ and, in general, each parameter can vary within a range of its own.

Fitness. Given two images C and D we take as fitness function their mutual information I, so the aim of the problem becomes to find the best transformation $\mathbf{T}$ for D such that the mutual information of C and $\mathbf{T}(D)$ is maximized.

4 The Distributed DE Algorithm

Our Distributed DE (DDE) algorithm is based on the classical coarse–grained approach to Evolutionary Algorithms, widely known in literature [20]. It consists of a locally connected topology of (in our case) DE instances, where each of them is connected to μ instances only. If, for example, we arrange them as a folded torus, then each DE instance has exactly four neighbouring populations.

Moreover every M_I generations (migration interval), neighbouring subpopulations exchange individuals. The percentage of individuals each population sends to its neighbours is called migration rate (M_R). All of the chosen individuals, be their number S_I, are sent to all of the neighbours, so each subpopulation receives a total number of $S_I \cdot \mu$ elements at each migration time.

Within this general framework we have implemented a parallel version for DE, which consists of a set of classical DE schemes (*slaves*), running in parallel, assigned to different processors arranged in a folded torus topology, plus a *master*. The pseudo-code of any slave process is delineated below.

Algorithm 1. Pseudocode of the DE slave

```
Procedure slave-DE
begin
    randomly generate an initial population of P individuals;
    evaluate goodness of each individual;
    while (termination criterion not fulfilled) do
        create the new population by classical DE actions;
        evaluate goodness of individuals in the new population;
        send the current best individual to the master process;
        if time to migrate
          send the current best individual to neighbouring populations;
          receive the best individuals from neighbouring populations;
          replace some local individuals with the received solutions;
        update variables for termination;
    od
end
```

The master process acts as an interface to the user: it simply collects the current local best solutions of the 'slave' processes and saves the best element at each generation. Furthermore, it compares this latter solution against the best found so far, and it saves the best among them and shows it to the user.

5 Experiments and Results

We have considered both *Mosaicking* and *Change Discovery* problems. The first accounts for the registration of two images of San Francisco area taken at the same time, while the second examines two images of an agricultural area near San Francisco taken at different times and looks for the changes in the area.

In a very preliminary set of experiments we used a sequential DE mechanism to face the two problems. In both cases results were encouraging, nonetheless the computation time was quite high (tens of minutes on a 1.5 GHz Pentium 4 depending on the value of n), which lead us to devote our attention to a distributed version. The DDE algorithm has been implemented in C language and communications take place via MPI. All the experiments have been effected on a Beowulf system made up by a cluster with 17 (1 master and 16 slaves) 1.5 GHz Pentium 4 nodes interconnected by a FastEthernet switch.

We have arranged the slaves in a $4 \cdot 4$ folded torus topology ($\mu = 4$), each DE procedure sends only its current best individual ($S_I = 1$) and this exchange takes place every $M_I = 5$ generations. The goodness of these choices has been confirmed by preliminary experiments effected. As regards the exploitation of the received solutions, these replace the worst four elements in each local population.

DE parameters have been set as follows: $n = 30$, $g = 200$, $CR = 0.5$ and $F = 0.5$. No preliminary tuning phase has been performed. It is important to remark here that, differently from some papers in literature about use of EAs

Table 1. Problem variable ranges

	a_{11}	a_{12}	a_{21}	a_{22}	b_1	b_2
min	0.500	-0.500	-0.500	0.500	-200.0	-200.0
max	1.500	0.500	0.500	1.500	200.0	200.0

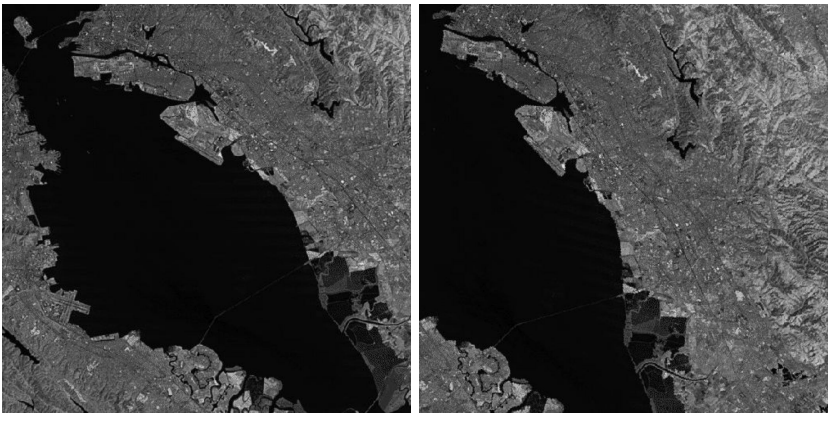


Fig. 1. The two original images for the *Mosaicking* task

to solve this task, we have decided to use quite wide ranges for each variable in the **T** solution, since we hope that evolution drive the search towards good transformations. The allowed variation ranges are shown in Tab. 1.

For each test problem 20 DDE executions have been carried out. The best of those runs will be described and discussed in the following in terms of image transformation achieved and evolution taken place.

The *Mosaicking* Task. In the first test case we have used two images which are manually selected portions of a Landsat Thematic Mapper (TM) digital image recorded on September 7, 1984 over San Francisco bay area (CA, USA) (property of United States Geological Survey [21]). Those images are color composites generated using Landsat TM spectral bands 2 (green), 4 (near-infrared), and 5 (mid-infrared) as blue, green, and red, respectively. Those images were transformed by us into grey monochannel images, so that each of them is $500 \cdot 500$ pixel large and uses 8 bits to represent each pixel. Figure 1 shows them both. Their I value is 0.1732. Figure 2 (top left) reports the fusion of the two original images. As it can be noticed, they share a common area, which should be used by the DDE algorithm to find their best registration. Namely, the up–left part of the second image overlaps the bottom–right part of the first, and a slight clockwise rotation was applied to the second image with reference to the first one. So, the best affine transformation should contain a slight counterclockwise rotation and two positive shifts for both the coordinates.

The best value of I obtained in the best execution is 1.1305. The average of the best final values over the 20 runs is 1.1299 and the variance is 0.0006,

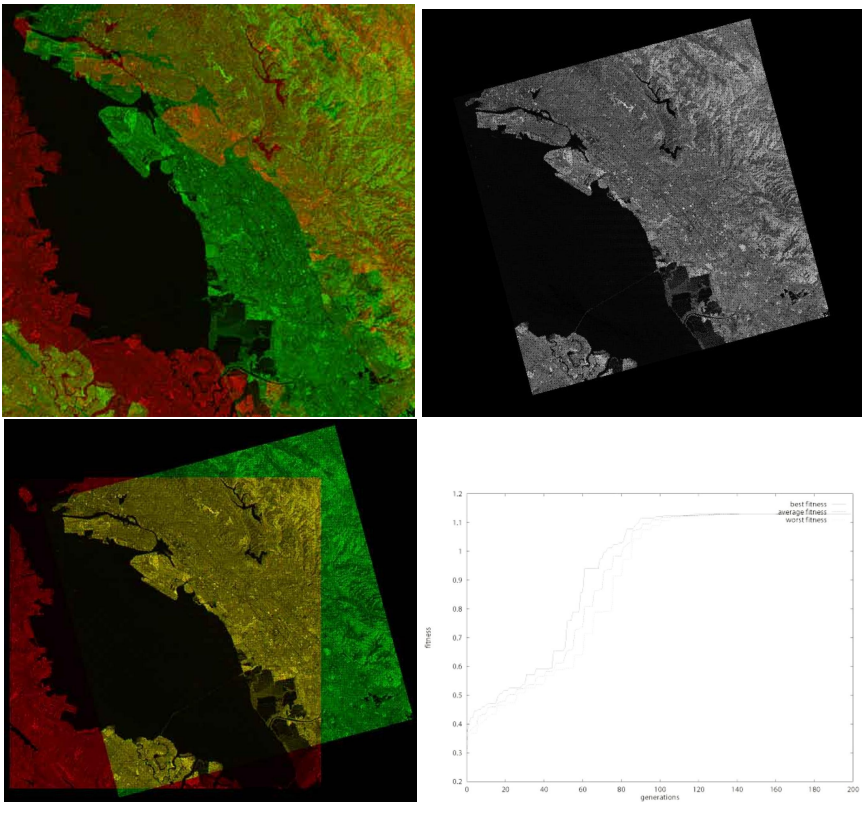


Fig. 2. Top Left: the fusion of the two original images. **Top Right:** The best transformation for the second *Mosaicking* image. **Bottom Left:** the first image is fused with the best transformation found for the second one. **Bottom Right:** Behavior of fitness as a function of the number of generations for the best run.

the worst result being 1.1278. The best affine transformation found is: $\mathbf{T} = \{0.946, -0.253, 0.253, 0.946, 42.141, 49.811\}$ which represents a counterclockwise rotation of about 15 degrees coupled with a translation in both axes. The resulting transformed image is shown in Fig. 2 (top right), while Fig. 2 (bottom left) depicts its fusion with the first image. The alignment of the two images is excellent: any detail in the first image, from streets to shorelines to bridges, is perfectly aligned with the corresponding pixels in the second one.

In Fig. 2 (bottom right) we report the evolution of the best run achieved for the *Mosaicking* task. Namely, we report the best, average and worst fitness values among those sent to the master by the 16 slaves at each generation. In spite of the very relaxed parameter range allowed, already the initial population achieves an improving solution with respect to the original one. From then on the system proposes many improving affine transformations and the average, the best and the worst fitness values increase over generations until the end of the run. It must be remarked that the values for the three above mentioned fitnesses

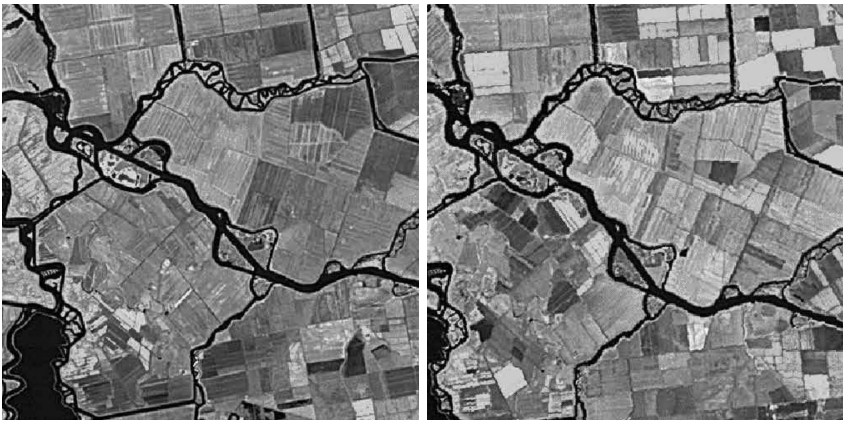


Fig. 3. The two original images for the *Change Discovery* task

are different one another until generation 196, though this is quite difficult to see from the figure especially after generation 140. This implies that good solutions spread only locally among linked subpopulations without causing premature convergence to the same suboptimal solution on all the slaves.

The *Change Discovery* Task. In the second test case we have used two images which refer to about the same area but were taken at different times. Namely, they represent an agricultural area near San Francisco (CA, USA) in 1984 and in 1993 respectively (they too are property of USGS [21]). As before, the original Landsat TM images were transformed by us into grey monochannel images, so that each of them is $500 \cdot 500$ pixel large with an 8–bit representation for each pixel (see Fig. 3). Their I value is 0.1123. Figure 4 (top left) reports the fusion of the two original images. As it can be observed, they share a common area, which should be used by the DDE algorithm to find their best registration. Namely, the right part of the first image overlaps the left part of the second, and a slight clockwise rotation took place when the second image was taken with reference to the first one. So, the best affine transformation should contain a slight counterclockwise rotation and some shifts for both the coordinates.

The best value of I attained in the best execution is 0.3961. The average of the best final values over the 20 runs is 0.3959 and the variance is 0.0001, the worst result being 0.3957. The best affine transformation found is: $\mathbf{T} = \{0.954, -0.083, 0.083, 0.953, 16.981, 20.282\}$ which represents a counterclockwise rotation of about 5 degrees coupled with a translation in both axes. The resulting transformed image is shown in Fig. 4 (top right), while Fig. 4 (bottom left) shows its fusion with the first image. The alignment of the two images is very good: any detail in the first image, from rivers to roads to fields, is well aligned with the corresponding pixels representing it in the second one.

Figure 4 (bottom right) presents the evolution of the best run achieved for the *Change Discovery* task. Also in this case we report the best, average and worst

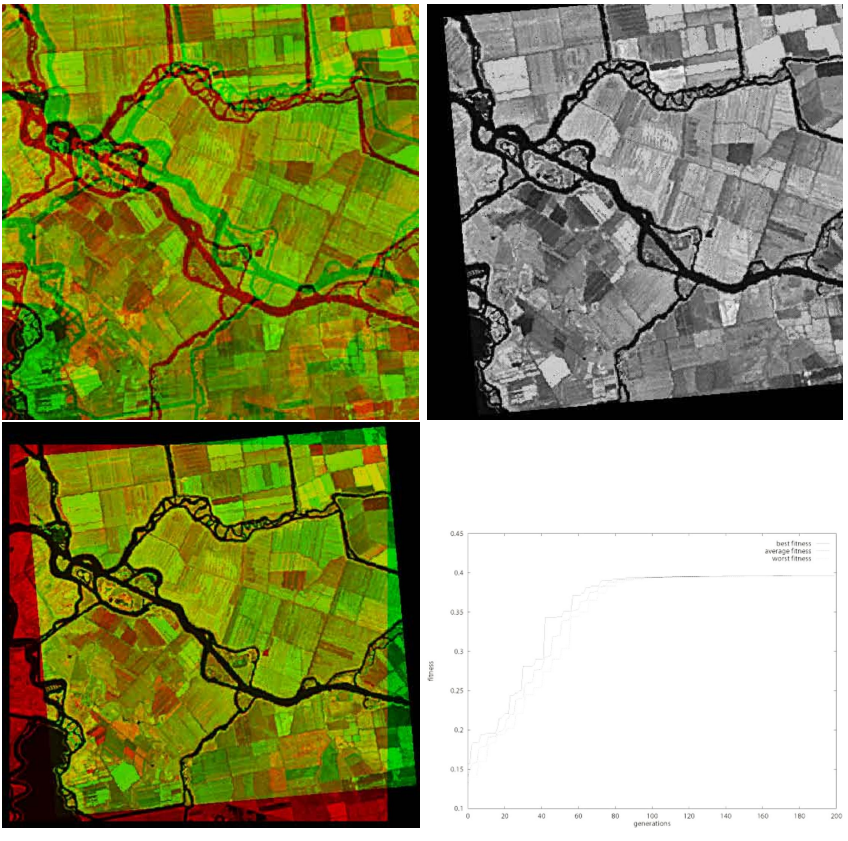


Fig. 4. Top Left: the fusion of the two original images.**Top Right:** The best transformation for the second *Change Discovery* image. **Bottom Left:** the first image is fused with the best transformation found for the second one. **Bottom Right:** Behavior of fitness as a function of the number of generations for the best run.

fitness values among those sent to the master by the 16 slaves at each generation. Also here, in spite of the very relaxed parameter range allowed, already the initial population achieves an improving solution with respect to the original one and from then on the system proposes many improving affine transformations and the fitness values increase over generations until the end of the run. The values for the three above mentioned fitnesses are different one another until the end of the run, though it is quite difficult to notice from the figure especially after generation 140. Thus considerations similar to those effected for the *Mosaicking* task about the behavior of our tool can be made in this case too.

The computed differences between the first image and the transformed second one are shown in Fig. 5, where only the part in which the two images overlap is meaningful. In it the grey color refers to areas where no changes occurred, the black represents areas burnt in 1984 and recovered by 1993, whereas the white stands for areas more vegetated in 1984 than in 1993 due to differences in the



Fig. 5. Change image in an agricultural area near San Francisco within 1984 and 1993

amount of rainfall, or to the density of the vegetation. Moreover, light pixels represent areas burned in 1993 or which were natural landscape areas in 1984 that were converted to agricultural lands and recently tilled, and finally dark pixels stand for areas more vegetated in 1993 than in 1984.

Speedup. In both problems we need to compare the time spent by DDE (t_{DDE}, of about 10 minutes) against the time t_{seq} of a sequential version using a population equal to the total number of individuals in the distributed algorithm, i.e. $16 \cdot 30 = 480$. This results in a speedup $s = t_{seq}/t_{DDE} = 8060/640 = 12.59$ and the related efficiency is $\epsilon = s/17 = 0.74$.

6 Conclusions and Future Works

In this paper a distributed version of the Differential Evolution strategy has been coupled with affine transformation and Mutual Information maximization to perform registration of remotely sensed images without any need of control points. A cluster of 17 personal computers has been used. The results seem to imply that this approach is promising, yet there is plenty of work still to do. Therefore, future works shall aim to shed light on the effectiveness of our system in this field, and on its limitations as well. A wide tuning phase shall be carried out to investigate if some DE parameter settings are, on average, more useful than others and to analyze the influence of the parameters on performance. This phase shall take into account lots of image couples taken from different fields. A comparison must be carried out against the results achieved by other image registration methods, to examine the effectiveness of our proposed approach.

References

1. Brown L G (1992) A Survey of Image Registration Techniques. ACM Computing Surveys 24(4): 325–376.
2. Zitova B, Flusser J (2003) Image Registration Methods: A Survey. Image and Vision Computing, 21:977–1000.
3. Ton J, Jain A K (1989) Registering Landsat Images by Point Matching. IEEE Trans. on Geoscience and Remote Sensing 27 (5):642–651.
4. Fonseca L M G, Manjunath B S (1996) Registration Techniques for Multisensor Remotely Sensed Imagery. Photogrammetric Engineering & Remote Sensing, 62(9): 1049–1056.
5. Lee C, Bethel J (2001) Georegistration of Airborne Hyperspectral Image Data. IEEE Trans. on Geoscience and Remote Sensing 39 (7):1347–1351.
6. Hart G W, Levy S, McLenaghan R (1995) Geometry. In: Zwillinger D (Ed), CRC Standard Mathematical Tables and Formulae. CRC Press, Boca Raton, FL.
7. Goldberg D (1989) Genetic Algorithms in Optimization, Search and Machine Learning. Addison Wesley, New York.
8. Eiben A E, Smith J E (2003) Introduction to Evolutionary Computing. Springer.
9. Fitzpatrick J, Grefenstette J, Gucht D (1984) Image Registration by Genetic Search. In: Proc. of the IEEE Southeast Conf., pp. 460–464.
10. Dasgupta D, McGregor D R (1992) Digital Image Registration using Structured Genetic Algorithms. In: Proceedings of SPIE The Int. Society for Optical Engineering, vol. 1776, pp. 226–234.
11. Chalermwat P, El-Ghazawi T A (1999) Multi-Resolution Image Registration Using Genetics. In: Proc. of the Int. Conf. on Image Processing, vol. 2, pp. 452–456.
12. Kim T, Im Y (2003) Automatic Satellite Image Registration by Combination of Stereo Matching and Random Sample Consensus. IEEE Trans. on Geoscience and Remote Sensing, 41(5):1111–1117.
13. Storn R, Price K (1997) Differential Evolution - a Simple and Efficient Heuristic for Global Optimization over Continuous Spaces. Journal of Global Optimization 11(4):341–359. Kluwer Academic Publishers.
14. Price K, Storn R, Lampinen J (2005), Differential Evolution: A Practical Approach to Global Optimization, Natural Computing Series. Springer-Verlag.
15. Dai X, Khorram S (1999) A Feature-based Image Registration Algorithm using Improved Chain-code Representation Combined with Invariant Moments. IEEE Trans. on Geoscience and Remote Sensing 37(5):2351–2362.
16. Thomas P, Vernon D (1997) Image Registration by Differential Evolution. In: Proc. of the Irish Machine Vision and Image Processing Conf., pp. 221–225, Magee College, University of Ulster, Ireland.
17. Netanyahu N S, Le Moigne J, Masek J G (2004) Georegistration of Landsat Data via Robust Matching of Multiresolution Features. IEEE Transactions on Geoscience and Remote Sensing 42:1586–1600.
18. Maes F, Collignon A, Vandermeulen D, Marchal G, Suetens P (1997) Multimodality Image Registration by Maximization of Mutual Information. IEEE Trans. on Medical Imaging 16(2):187–198.
19. Pluim J P W, Maintz A J B, Viergever M A (2003) Mutual–information–based Registration of Medical Images: a Survey. IEEE Trans. on Medical Imaging 22:986–1004.
20. Cantú–Paz E (1995) A Summary of Research on Parallel Genetic Algorithms. *IlliGAL Report*, no. 95007, University of Illinois at Urbana–Champaign, USA.
21. http://terraweb.wr.usgs.gov/projects/SFBay/

Harmonic Estimation Using a Global Search Optimiser

Y.N. Fei[1], Z. Lu[2], W.H. Tang[2], and Q.H. Wu[2,⋆]

[1] School of Engineering and Technology, Shenzhen University,
Shenzhen 518060, P.R.China
[2] Department of Electrical Engineering and Electronics, The University of Liverpool,
Liverpool L69 3GJ, U.K.
qhwu@liv.ac.uk

Abstract. Accurate harmonic estimation is the foundation to ensure a reliable power quality environment in a power system. This paper presents a new algorithm based on a Group Search Optimiser (GSO) to estimate the harmonic components presented in a voltage or current waveform. The structure of harmonic estimation is represented as linear in amplitude and non-linear in phase. The proposed algorithm takes advantage of this feature and estimates amplitudes and phases of harmonics by a linear Least Squared (LS) algorithm and a non-linear GSO-based method respectively. The improved estimation accuracy is demonstrated in this paper in comparison with that of the conventional Discrete Fourier Transform (DFT) and Genetic Algorithms (GAs). Moreover, the performance is still satisfactory even in simulations with the presence of inter-harmonics and frequency deviation.

Keywords: Group Search Optimiser (GSO), Discrete Fourier Transform (DFT), Genetic Algorithm (GA), inter-harmonics.

1 Introduction

With the increasing use of power-electronic equipment in power systems, harmonic pollution, produced by non-linear electronically controlled equipment, has significantly deteriorated the power quality in electrical power networks. It is therefore necessary to estimate the harmonic components in power systems for the assessment of the quality of the power delivered. The assessment refers to estimating parameters of the harmonics, such as the amplitudes and phases of the corresponding harmonic components.

There are various approaches to estimate the harmonic parameters of an electrical signal [1][2]. The most widely used ones are fast executable algorithms derived from the Discrete Fourier Transform (DFT). However, there are three major pitfalls in the applications of the DFT-based algorithms, *i.e.*, aliasing, leakage and picket fence phenomena, if certain undesirable conditions are given. The Kalman filtering approach has also been frequently employed to estimate

⋆ Corresponding author: Tel: +44 1517944535, Fax: +44 1517944540.

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 261–270, 2007.
© Springer-Verlag Berlin Heidelberg 2007

different states and parameters of harmonics in an electrical signal [1]. It utilises simple, linear and robust algorithms to estimate the magnitudes of the harmonics embedded in the electrical signal. However, the Kalman filtering approach requires a prior knowledge of the statistics of the electrical signal and the state matrix needs to be defined accurately as well.

Stochastic optimisation methods, such as Genetic Algorithms (GAs), have been developed in the past a few decades. The main attraction of the GA lies in the fact that it does not rely on Newton-like gradient descent methods. The derivatives of the cost function are not required during optimisation that makes the search less likely to be trapped in local minima. However, the efficiency of GA is significantly degraded when it is applied to highly epistatic objective functions, *e.g.*, in a function where the parameters being optimised are highly correlated [3][4]. Since the parameters of each harmonic are correlative in real electrical signals, the performance of the GA is affected due to the premature convergence. Hence, its search capability is limited. Group Search Optimiser (GSO) is a novel stochastic optimisation algorithm which was inspired by animal searching behaviour and group living theory [5]. Based on a Producer-Scrounger model [6], GSO provides an open framework to utilise research in animal behavioural ecology to solve optimisation problems. GSO has competitive performance to other stochastic optimisation methods in terms of accuracy and convergence speed. With only one parameter which needs to tune, GSO is attractive in an implementation point of view.

In this paper, the GSO algorithm is proposed as a new solution for the estimation of magnitudes and phases of harmonics. First of all, the basic theory of GSO is reported. Then, the algorithm is implemented to estimate the phases of fundamental and harmonic components, followed by the discussions in the field of inter-harmonic estimation and in the cases of frequency deviation. The advantages of the proposed algorithm are also demonstrated in comparison with GA and DFT. The conclusions are drawn at the end of this paper.

2 Global Search Optimiser

Swarm intelligence is a nature-inspired computational intelligence technique based upon the study of collective behaviour in decentralised, self-organised systems, *e.g.*, a group of animals. In nature, the resource searching process of animals is analogous to optimisation that is a process of seeking optima in a search space. GSO is developed by drawing inspiration from animal searching behaviour, especially group searching behaviour. The Producer-Scrounger (PS) model, associated with concepts of resource searching from animal scanning mechanisms, is employed as a framework to design optimum searching strategies for GSO.

The PS model was proposed to analyse social foraging strategies of group living animals in [6]. Two foraging strategies are used within groups. The first one is producing, *e.g.*, searching for food; the other one is joining or called scrounging, *e.g.*, joining resources uncovered by others. In the PS model, foragers are assumed to use producing or joining strategies exclusively.

Basically GSO is a population-based optimisation algorithm. The population of GSO is called a *group* and each individual in the population is a *member*. In a group, there are three kinds of members: producers, scroungers and rangers. The behaviours of producers and scroungers are based on the PS model, while rangers perform random walks to avoid entrapment in local minima in a search space. For the purpose of accuracy and convenience of calculation, only one member is appointed as a producer at each searching bout in the group, and the remaining members are scroungers and rangers. All the scroungers will join the resource found by the producer.

In an n-dimensional search space, the i_{th} member at the k_{th} searching bout (iteration) in a group, has a current position $X_i^k \in \mathbb{R}^n$, a head angle $\varphi_i^k = (\varphi_{i1}^k, \ldots, \varphi_{i(n-1)}^k) \in \mathbb{R}^{n-1}$ and a head direction $D_i^k(\varphi_i^k) = (d_{i1}^k, \ldots, d_{in}^k) \in \mathbb{R}^n$. The head direction can be calculated from the head angle φ_i^k via a Polar to Cartesian coordinate transformation:

$$\begin{aligned} d_{i1}^k &= \prod_{p=1}^{n-1} \cos(\varphi_{ip}^k) \\ d_{ij}^k &= \sin(\varphi_{i(j-1)}^k) \cdot \prod_{p=i}^{n-1} \cos(\varphi_{ip}^k) \\ d_{in}^k &= \sin(\varphi_{i(n-1)}^k) \end{aligned} \tag{1}$$

During each search bout, there always exists a member in the group which is located in the most promising area and conferred the best fitness value. This member is assigned as the producer. At intervals of each search bout, the producer scans the environment to search resources (optima). Scanning can be accomplished through physical contact or by visual, chemical, or auditory mechanisms [7]. Vision is used by the producer in GSO since it is the main scanning mechanism used by many animal species. In order to handle optimisation problems whose dimensions of parameters are larger than 3, the scanning field of vision is generalised to an n-dimensional space, which is characterised by maximum pursuit angle $\theta_{\max} \in \mathbb{R}^{n-1}$ and maximum pursuit distance $l_{\max} \in \mathbb{R}^1$ as illustrated in a 3D space as shown in Figure 1.

In the GSO algorithm, the producer X_p at the k_{th} iteration behaves as follows:

1) The producer will scan at zero degree and then scan aside by randomly sampling three points in the scanning field, one point at zero degree:

$$X_z = X_p^k + r_1 l_{\max} D_p^k(\varphi_p^k) \tag{2}$$

one point in the right hand side hypercube:

$$X_r = X_p^k + r_1 l_{\max} D_p^k(\varphi_p^k + r_2 \theta_{\max}/2) \tag{3}$$

and one point in the left hand side hypercube:

$$X_l = X_p^k + r_1 l_{\max} D_p^k(\varphi_p^k - r_2 \theta_{\max}/2) \tag{4}$$

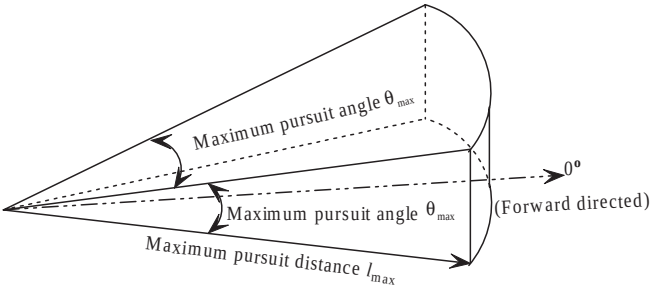


Fig. 1. Scanning field in a 3D space [7]

where $r_1 \in \mathbb{R}^1$ is a normally distributed random number with mean 0 and standard deviation 1, and $r_2 \in \mathbb{R}^{n-1}$ is a random sequence in the range $(0, 1)$.

2) If the producer can find the best point with a better resource (fitness value) than its current position, then it will fly to this point. Otherwise it stays in its current position and turns its head to a new angle:

$$\varphi^{k+1} = \varphi^k + r_2 \alpha_{\max} \tag{5}$$

where $\alpha_{\max}$ is the maximum turning angle and r_2 is the same as the one used in (4).

3) If the producer cannot find a better position after a iterations, it will turn its head back to zero degree:

$$\varphi^{k+a} = \varphi^k \tag{6}$$

where a is a constant given by $round(\sqrt{n+1})$.

A number of members in the group are selected as scroungers at each iteration. The scroungers keep searching for opportunities to join the resources found by the producer. In GSO, the commonest scrounging behaviour in house sparrows (*Passer domesticus*) [6], area copying, is adopted. The scroungers move across to search in the immediate area around the producer. At the k_{th} iteration, the behaviour of the i_{th} scrounger is modelled as a random walk towards the producer:

$$X_i^{k+1} = X_i^k + r_3(X_p^k - X_i^k) \tag{7}$$

where $r_3 \in \mathbb{R}^n$ is a uniform random sequence in the range $(0, 1)$.

Other members in the group are rangers. Rangers are introduced to explore a new search space therefore to avoid entrapment of local minima. Random walks are employed by rangers in GSO. At the k_{th} iteration, if the i_{th} member of in the group is selected as a ranger, it turns its head to a random head angle φ_i^{k+1} by:

$$\varphi_i^{k+1} = \varphi_i^k + r_2 \alpha_{\max} \tag{8}$$

where $\alpha_{\max}$ is the maximum turning angle; and it chooses a random walk distance:

$$l_i = a \cdot r_1 l_{\max} \tag{9}$$

So it will move to a new point:

$$X_i^{k+1} = X_i^k + l_i D_i^k(\varphi^{k+1}) \tag{10}$$

In order to maximise chances in finding resources, animals restrict their search in a profitable patch. When the edge of the search space is reached, they will turn back into a patch [8]. To handle an optimisation problem with a bounded search space, the following strategy is employed: when a member is outside the search space, it will turn back to its previous position inside the search space.

3 Harmonic Estimation

3.1 Estimation of Integral-Harmonics

This section presents the procedure of using GSO to estimate harmonics in an electrical signal. Assume a signal $Z(t)$ is described as below:

$$Z(t) = \sum_{n=1}^{N} A_n \sin(2n\pi f_0 t + \phi_n) + v(t), \tag{11}$$

where n, from 1 to N, represents the order of the harmonics to be estimated; A_n and ϕ_n are the amplitude and phase angle of the n^{th} harmonic; f_0 is the fundamental frequency of the signal; and $v(t)$ is the additive noise. Since the phases of the harmonics in the model are non-linear, GSO is utilised to estimate the values of ϕ_n. With the phases estimated in each iteration, the amplitudes of A_n are calculated by a standard Least Squared (LS) algorithm.

The discrete linear model of the signal $Z(t)$ with additive noise is given as:

$$Z(k) = H(k) \cdot A + v(k), \tag{12}$$

$$H(k) = \begin{bmatrix} \sin(w_1 t_1 + \phi_1) & \dots & \sin(w_n t_1 + \phi_n) \\ \sin(w_1 t_2 + \phi_1) & \dots & \sin(w_n t_2 + \phi_n) \\ \vdots & \dots & \vdots \\ \sin(w_1 t_k + \phi_1) & \dots & \sin(w_n t_k + \phi_n) \end{bmatrix}, \tag{13}$$

where $Z(k)$ is the k^{th} sample of the measured values with additive noise $v(k)$; $A = [A_1 \ A_2 \ \cdots \ A_N]^T$, which is the vector of the amplitudes that need to be estimated; $H(k)$ is the system structure matrix and the vectors of w_n in it have the values of $2n\pi f_0$ $(n = 1, 2, \cdots, N)$. The estimation model for the signal is:

$$\hat{Z}(k) = H(k)\hat{A}. \tag{14}$$

Assuming that $\hat{H}(k)$ is a full-rank matrix, the estimation of $\hat{A}$ is obtained via the standard LS algorithm as:

$$\hat{A} = [\hat{H}^T(k) \cdot \hat{H}(k)]^{-1} \hat{H}^T(k) Z(k). \tag{15}$$

With the phases and amplitudes estimated, $\hat{Z}$ can be calculated by the estimation model in (14). The performance of GSO is evaluated using both the amplitudes and phases to calculate a cost function J. The process is repeated until the final convergence condition is reached. The cost function J for a time window T is calculated as:

$$J = \int_0^T (Z(t) - \hat{Z}(t))^2. \tag{16}$$

Given a data set which is sampled from a section of voltage or current waveform contaminated with additive noise, a number of parameters need to be initialised, including: N, the number of parameters need to be optimised, which denotes the number of phases to be estimated; P, the number of particles to be used for searching; and G, the generation of the update of particles in GSO to be repeated. A population of particles (phases) is randomly selected in the searching space. The i^{th} particle at iteration k in the population is denoted as:

$$X_i^k = (\phi_1^k, \phi_2^k, \cdots, \phi_N^k). \tag{17}$$

For each particle, calculate $H(k)$ by (13) and estimate the amplitudes $\hat{A}$ by (15), then the cost function J is used to evaluate the performance of each particle. Based on the values of the cost function, the best particle is found and the group of particles is updated according to the process described in the preceding sections. These steps are repeated until a maximum predefined number of the generations T is reached or the value of the cost function of the best particle is minimised to a small value ε.

3.2 Estimation of Inter-harmonics and Sub-harmonics

Inter-harmonics are spectral components at frequencies that are not integer multiple of the system fundamental frequency in signals. Sub-harmonics are inter-harmonics with frequencies lower than the fundamental frequency. The main sources of inter-harmonics and sub-harmonics are electronic devices, such as cyclo-converters, arc furnaces and integral cycle controlled furnaces. The presence of inter-harmonics and sub-harmonics significantly increases difficulties both in modelling and measurement of signals.

A number of algorithms have been proposed for harmonic analysis, and DFT is the most widely used computation algorithm. However, leakage effect, picket-fence effect, and aliasing effect make DFT suffer from specific restrictions. The spectral leakage problem is originated from two main causes in harmonic analysis. One is the presence of inter-harmonics which causes that the analysed time length of the signal is not synchronised with the length of DFT in the calculation; and another is the deviation of the fundamental frequency value.

The presence of inter-harmonics introduces large errors in harmonic estimation using DFT. However, if prior knowledge of the frequencies of the inter-harmonics is given, a high accuracy of harmonic estimation can be achieved using the proposed GSO in a fixed sampling window. Let ϕ_{i1} and ϕ_{i2} represent

the phase angles of a sub-harmonic and an inter-harmonic component respectively. The dimension of the system structure matrix $H(k)$ is increased from $n \times k$ to $(n+2) \times k$ and the position of the i^{th} particle at iteration k is represented as:

$$X_i^k = (\phi_{i1}^k, \phi_1^k, \phi_{i2}^k \cdots, \phi_N^k). \tag{18}$$

Accordingly, the search space is extended to $(n+2)$-dimension.

3.3 Estimation of Frequency Deviation

For the analysis of harmonics, frequency deviation is another factor affecting the accuracy of DFT. The frequency of harmonics are the multiple of the fundamental frequency. So, if an accurate value of the fundamental frequency is obtained, the frequencies for each harmonic can be calculated as well. Therefore, the key point of harmonic estimation is how to estimate the fundamental frequency.

The procedure of the fundamental frequency estimation, using GSO, is discussed as follows: Add the fundamental frequency as a parameter into each particle of the population, then the search space is increased to $(n+1)$-dimension. The position of the i_{th} particle at iteration k is given by:

$$X_i^k = (\phi_1^k, \cdots, \phi_N^k, f_0^k). \tag{19}$$

The estimated fundamental frequency is updated in each iteration of GSO until the final convergence condition is reached. In addition to the phases and amplitudes derived, the analysed electrical signal can be extracted or reconstructed with the parameters of all the harmonics estimated.

4 Simulation Results

For the purpose of comparison, a simulation module is employed to produce a test signal, which is as same as the one used in [9]. Figure 2 describes a simple power system, which is comprised of a two-bus three-phase system with a full-wave six-pulse bridge rectifier at the load bus. The test signal is a distorted voltage waveform sampled from the terminal of the load bus in the simple power system. The frequencies and phases of the harmonics of the test signal are listed in Table 1. The test signal is sampled 64 points per cycle from a 50 Hz voltage waveform. The algorithm is operated under no noise and noisy situations, respectively. The Signal to Noise Rations (SNRs) at 20, 10 and 0 dB are chosen in simulations under noisy situations.

The number of harmonics considered in the test signal for this simulation case is five. The system matrix H is given in (13). The number of parameters to be estimated in this case is ten: five phases and five amplitudes for the predefined fundamental, 5^{th}, 7^{th}, 11^{th} and 13^{th} harmonics. A waveform can be reconstructed by the harmonics estimated using the values of these parameters. In the simulations, the performance index, %error is estimated by:

$$\%\text{error} = \left[\int_0^T (Z(t) - \hat{Z}(t))^2 / \int_0^T Z(t)^2 \right] \times 100. \tag{20}$$

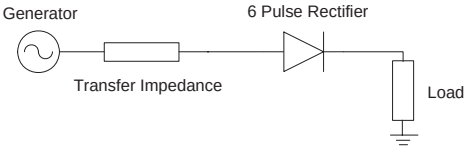


Fig. 2. A simple power system-a two-bus architecture with six-pulse full-wave bridge rectifier supplying the load

Table 1. Harmonic content of the test signal

Harmonic Order	Amplitude (P.U.)	Phase (Degrees)
Fundamental (50 Hz)	0.95	-2.02
5^{th}(250 Hz)	0.09	82.1
7^{th}(350 Hz)	0.043	7.9
11^{th}(550 Hz)	0.03	-147.1
13^{th}(650 Hz)	0.033	162.6

The estimation results of the proposed GSO scheme in comparison with DFT and GA, are given in Table 2. GA uses the same structure, population and generations as the proposed GSO scheme does in the study. In Table 2, it is a significant improvement in terms of reducing the errors for the harmonic estimation using GSO in comparison with DFT. With the same generation of population used, GSO achieved an improved accuracy compared with GA.

Table 2. Comparison of the errors in GSO, GA and DFT for the estimation of all the harmonics

	Uniform noise			Gaussian noise		
SNR	DFT %error	GA %error	GSO %error	DFT %error	GA %error	GSO %error
No Noise	0.2175	0.1651	5.4865×10^{-10}	0.2175	0.1651	5.4865×10^{-10}
20 dB	1.0894	0.8735	0.8292	2.1747	1.7634	1.7414
10 dB	6.4348	6.1459	6.1264	17.6317	16.3732	16.1902
0 dB	40.5646	39.4068	39.3195	66.2641	60.5760	60.4985

4.1 Measurement of Inter-harmonics

To evaluate the performance of the proposed GSO algorithm in the estimation of a signal in presence of inter-harmonics, a test signal $Z(t)$, with two inter-harmonics, is simulated. The frequencies of the two inter-harmonics are denoted as f_{i1} and f_{i2}, respectively. In this case, f_{i1} equals 20 Hz and f_{i2} equals 80 Hz. The amplitudes of the two inter-harmonics are A_{i1} and A_{i2}, which equal 0.505 P.U. and 0.185 P.U. The phases of the two inter-harmonics are 75.6 and -135.5 degrees respectively in this simulation.

Table 3. Comparison of the errors in GSO, GA and DFT for the estimation of all the harmonics including inter-harmonics

	Uniform noise			Gaussian noise		
SNR	DFT %error	GA %error	GSO %error %error	DFT %error	GA %error	GSO
No Noise	4.4515	0.1286	1.5381×10^{-4}	4.4515	0.1286	1.5381×10^{-4}
20 dB	6.6556	0.4925	0.0683	6.2340	0.7621	0.7086
10 dB	12.4357	0.9245	0.6397	12.0909	8.6969	7.9996
0 dB	41.2282	5.6261	5.1862	44.0044	38.0932	37.9452

DFT, GA and GSO are used for the estimation of the signal respectively. It can be concluded from Table 3 that the interference of inter-harmonics brings a large error to the measurement with DFT. It is ascertained that at a noisy condition, there is a significant improvement in performance in the case of GSO as compared with DFT in terms of reducing the %error. When compared with GA, GSO achieved an improved performance in all cases.

4.2 Frequency Deviation

In this subsection, the simulation test is presented for the purpose of comparing the proposed GSO scheme with DFT and a method named the Combined Method (CM) which was given in [10]. In this case, the fundamental frequency is deviated to 60.5 Hz and the sampling frequency is $60 \times 16 = 960$ Hz. The harmonic contents and the simulation results are presented in Table 4, and the estimated fundamental frequency by GSO is 60.5108 Hz. It should be noted that the frequency of the 8^{th} harmonic is $60.5 \times 8 = 484$ Hz, which is over the half of the sampling frequency 480 Hz. According to the Nyquist sampling theorem, this part of the signal cannot be recovered from the samples. This aliasing effect makes the estimation results of DFT deviate considerably from target values, as shown in Table 4. It demonstrates that the proposed GSO and CM gain more accurate values of the amplitudes of harmonics, while GSO has the best performance among the three.

Table 4. Simulation signal and results ($f_0 = 60.5$ Hz with 0.5% noise)

Harmonic Order	Amplitude	FFT	CM	GSO
1	1	1.0061	0.9998	1.0001
2	0.07	0.0373	0.0704	0.0705
3	0.05	0.0481	0.0500	0.0498
5	0.04	0.0384	0.0402	0.0399
7	0.03	0.0264	0.0293	0.0300
8	0.01	0.0152	0.0103	0.0097

5 Conclusion

This paper presents a new algorithm which can estimate harmonics accurately for power quality monitoring. The structure of distorted signals in power systems is decoupled into linear and non-linear parts. A GSO-based method is applied to estimate the non-linear parameters, *i.e.*, the phase of each harmonic and the fundamental frequency of the system. The linear parameters, the amplitudes, are then obtained using the well-known LS estimator.

In this study, GSO achieves an improved performance over the conventional GA and DFT in the presence of noise. Moreover, it does not suffer the drawbacks such as leakage and picket-fence effects which are the side-effects of the conventional DFT algorithm. Hence, the proposed algorithm is able to be applied for the estimation of harmonics even in the presence of inter-harmonics and frequency deviation. The estimation results obtained can provide accurate information for component design, operation and protection purposes in power systems.

References

1. Ma, H., Girgis, A.A.: Identification and tracking of harmonic sources in a power system using kalman filter. IEEE Transactions on Power Delivery **11** (Dec., 1996) 1659–1665
2. Girgis, A., Chang, W.B., Makram, E.B.: A digital recursive measurement scheme for on-line tracking of power system harmonics. IEEE Transactions on Power Delivery **6**(3) (1998) 1153–1160
3. Fogel, D.B.: Evolutionary Computation Toward a New Philosophy of Machine Intelligence. IEEE, New York (1995)
4. Mishra, S.: Optimal design of power system stabilizers using particle swarm optimization. IEEE Transactions on Energy Conversion **17**(3) (Sep., 2002) 406–413
5. He, S., Wu, Q.H., Saunders, J.R.: A novel group search optimizer inspired by animal behavioural ecology. In: 2006 IEEE Congress on Evolutionary Computation (CEC 2006), Sheraton Vancouver Wall Centre, Vancouver, BC, Canada. (July 2006) Tue PM–10–6
6. Barnard, C.J., Sibly, R.M.: Producers and scroungers: a general model and its application to captive flocks of house sparrows. Animal Behaviour **29** (1981) 543–550
7. Bell, J.W.: Searching Behaviour - The Behavioural Ecology of Finding Resources. Chapman and Hall (1990)
8. Dixon, A.F.G.: An experimental study of the searching behaviour of the predatory coccinellid beetle adalia decempunctata. Journal of Animal Ecology **28** (1959) 259–281
9. Bettayeb, M., Qidwai, U.: Recursive estimation of power system harmonics. Electric Power Systems Research **47** (1998) 143–152
10. Yang, J.Z., Yu, C.S., Liu, C.W.: A new method for power signal harmonic analysis. IEEE Transactions on Power Delivery **20**(2) (Apr., 2005)

An Online EHW Pattern Recognition System Applied to Face Image Recognition

Kyrre Glette[1], Jim Torresen[1], and Moritoshi Yasunaga[2]

[1] University of Oslo, Department of Informatics,
P.O. Box 1080 Blindern, 0316 Oslo, Norway
{kyrrehg,jimtoer}@ifi.uio.no
[2] University of Tsukuba, Graduate School of Systems and Information Engineering,
1-1-1 Ten-ou-dai, Tsukuba, Ibaraki, Japan
yasunaga@cs.tsukuba.ac.jp

Abstract. An evolvable hardware (EHW) architecture for high-speed pattern recognition has been proposed. For a complex face image recognition task, the system demonstrates (in simulation) an accuracy of 96.25% which is better than previously proposed EHW architectures. In contrast to previous approaches, this architecture is designed for online evolution. Incremental evolution and high level modules have been utilized in order to make the evolution feasible.

1 Introduction

Image recognition systems requiring a low recognition latency or high throughput could benefit from a hardware implementation. Furthermore, if the systems are applied in time-varying environments, and thus need adaptability, online evolvable hardware (EHW) would seem to be a promising approach [1].

One approach to online reconfigurability is the Virtual Reconfigurable Circuit (VRC) method proposed by Sekanina in [2]. This method does not change the bitstream to the FPGA itself, rather it changes the register values of a circuit already implemented on the FPGA, and obtains virtual reconfigurability. This approach has a speed advantage over reconfiguring the FPGA itself, and it is also more feasible because of proprietary formats preventing direct FPGA bitstream manipulation. However, the method requires much logic resources.

Experiments on image recognition by EHW were first reported by Iwata et al in [3]. A field programmable logic array (FPLA) device was utilized for recognition of three different patterns from black and white input images of 8x8 pixels. An EHW road image recognition system has been proposed in [4]. A gate array structure was used for categorizing black and white input images with a resolution of 8x4 pixels. Incremental evolution was applied in order to increase the evolvabiliy.

A speed limit sign recognition system has been proposed in [5]. The architecture employed a column of AND gates followed by a column of OR gates, and then a selector unit. A maximum detector then made it possible to decide a speed limit from 6 categories. Incremental evolution was applied in two ways: each subsystem was first evolved separately, and then in a second step the subsystems

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 271–280, 2007.
© Springer-Verlag Berlin Heidelberg 2007

were assembled and the selector units were evolved. The input images were black and white and had a resolution of 7x5 pixels.

An EHW face image classifier system, LoDETT, has been presented by Yasunaga et al. [6]. This system is capable of classifying large input vectors into several categories. For a face image recognition task, the input images had a resolution of 8x8 pixels of 8-bit grayscale values, belonging to 40 different categories. In this architecture, the classifier function is directly coded in large AND gates. The classification is based on detecting the category with the highest number of activated AND gates. Incremental evolution is utilized for this system too, where each module for detecting a category is evolved separately. The average recognition accuracy is 94.7% However, evolution is performed *offline* and the final system is synthesized. This approach gives rapid classification in a compact circuit, but lacks run-time reconfigurability.

The system we have developed earlier [7] addresses the reconfigurability by employing a VRC-like array of high-level functions. Online/on-chip evolution is attained, and therefore the system seems suited to applications with changes in the training set. However, the system is limited to recognizing one category out of ten possible input categories. The system uses the same image database as [6] with the same input resolution.

The architecture proposed in this paper expands to categorization of all 40 categories from the image database used in [6], while maintaining the on-line evolution features from [7]. A change in the architecture has been undertaken to accomodate for the recognition of multiple categories. While in LoDETT a large number of inputs to the AND gates could be optimized away during circuit synthesis, the run-time reconfiguration aspect of the following architecture has led to a different approach employing fewer elements.

A large amount of litterature exists on conventional face image recognition. A comprehensive survey can be found in [8]. Work on conventional hardware face recognition has been undertaken, based on the modular PCA method [9]. However, the recognition speed (11 ms) is still inferior to the LoDETT system.

The next section introduces the architecture of the evolvable hardware system. Aspects of evolution are discussed in section 3. Results from the experiments are given and discussed in sections 4. Finally, section 5 concludes the paper.

2 The EHW Architecture

The EHW architecture is implemented as a circuit whose behaviour and connections can be controlled through configuration registers. By writing the genome bitstream from the genetic algorithm to these registers, one obtains the phenotype circuit which can then be evaluated. This approach is related to the VRC technique, as well as to the architectures in our previous works [10,7].

2.1 System Overview

The classifier system consists of K category detection modules (CDMs), one for each category C_i to be classified – see figure 1. The input data to be classified are

presented to each CDM concurrently on a common input bus. The CDM with the highest output will be detected by a maximum detector, and the number of this category will be output from the system. Alternatively, the system could also state the degree of certainty of a certain category by taking the output of the corresponding CDM and dividing by the maximum possible output. In this way, the system could also propose alternative categories in case of doubt.

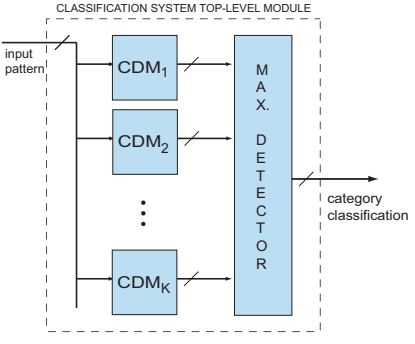


Fig. 1. EHW classifier system view. The pattern to be classified is input to all of the category detection modules.

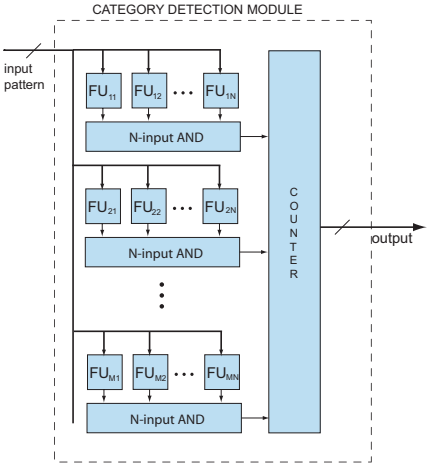


Fig. 2. Category detection module. N functional units are connected to an N-input AND gate.

2.2 Category Detection Module

Each CDM consists of M "rules" or functional unit (FU) rows. See figure 2. Each FU row consists of N FUs. The inputs to the circuit are passed on to the inputs of each FU. The 1-bit outputs from the FUs in a row are fed into an N-input AND gate. This means that all outputs from the FUs must be 1 in order for a rule to be activated. The outputs from the AND gates are connected to an input counter which counts the number of activated FU rows.

2.3 Functional Unit

The FUs are the reconfigurable elements of the architecture. As seen in figure 3, each FU behavior is controlled by configuration lines connected to the configuration registers. Each FU has all input bits to the system available at its inputs, but only one data element (e.g. one byte) of these bits is chosen. One data element is thus *selected* from the input bits, depending on the configuration lines. This data is then fed to the available functions. The choice of functions for

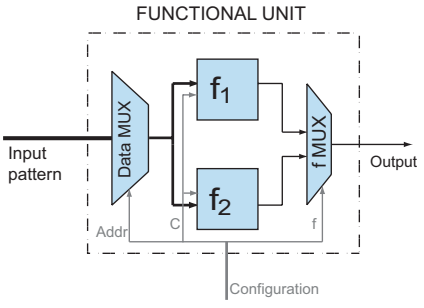


Fig. 3. Functional unit. The configuration lines are shown in gray. The data MUX selects which of the input data to feed to the functions f_1 and f_2. The constant C is given by the configuration lines. Finally, the f MUX selects which of the function results to output.

this application will be detailed in section 2.4. In addition, the unit is configured with a constant value, C. This value and the input byte are used by the function to compute the output from the unit.

The advantage of selecting which inputs to use, is that one is not required to connect to all inputs. A direct implementation of the LoDETT system [6] would have required, in the image recognition case, $N = 64$ FUs in a row. Our system typically uses $N = 6$ units. The rationale is that not all of the inputs are necessary for the pattern recognition. This is reflected in the *don't care*s evolved in [6].

2.4 Face Image Recognition

The pattern recognition system has been applied to face image recognition. The fitness of the face recognition system is based on the system's ability to recognize the correct person from a range of different face images. The images are taken from the AT&T Database of Faces (formerly "The ORL Database of Faces")[1] which contains 400 images divided into 40 people with 10 images each. For each person, images are taken with variations such as different facial expressions and head tilt. The original resolutions of the images were 92x112 pixels, 8-bit grayscale. In our experiment the images were preprocessed by a downsampling to 8x8 pixels, 8-bit grayscale. This was done to reduce noise and the number of inputs to the system. The input pattern to the system is then 64 pixels of 8 bits each (512 bits in total).

Based on the data elements of the input being 8-bit pixels, the functions available to the FU elements have been chosen to *greater than* and *less than*. Through experiments these functions have shown to work well, and intuitively this allows for detecting dark and bright spots. Combined use of these functions for the same pixel makes it possibe to define an intensity range. The constant is also 8 bits, and the input is then compared to this value to give *true* or *false*

[1] http://www.cl.cam.ac.uk/Research/DTG/attarchive/facedatabase.html

as output. This can be summarized as follows, with I being the selected input value, O the output, and C the constant value:

f	Description	Function
0	Greater than	$O = 1$ if $I > C$, else 0
1	Less than	$O = 1$ if $I < C$, else 0

3 Evolution

This section describes the evolutionary process. The genetic algorithm (GA) implemented for the experiments follows the Simple GA style [11]. The algorithm is written to be run on an embedded processor, such as the PowerPC 405 core in Xilinx Virtex-II Pro or better FPGAs [10]. Allowing the GA to run in software instead of implementing it in hardware gives an increased flexibility compared to a hardware implementation.

The GA associates a bit string (genome) with each individual in the population. For each individual, the EHW circuit is configured with the associated bit string, and training vectors are applied on the inputs. By reading back the outputs from the circuit, a fitness value can be calculated.

3.1 Genome

The encoding of each FU in the genome string is as follows:

Pixel address (6 bit)	Function (1 bit)	Constant (8 bit)

This gives a total of $B_{unit} = 15$ bits for each unit. The genome for one FU row is encoded as follows:

FU_1(15b)	FU_2(15b)	...	FU_N(15b)

The total amount of bits in the genome for one FU row is then, with $N = 8$, $B_{tot} = B_{unit} \times N = 15 \times 8 = 120$.

3.2 Incremental Evolution of the Category Detectors

Evolving the whole system in one run would give a very long genome, therefore an incremental approach is chosen. Each category detector CDM_i is evolved separately, since there is no interdependency between the different categories. This is also true for the FU rows each CDM consists of. Thus, the evolution can be performed on one FU row at a time. This significally reduces the genome size.

One then has the possibility of evolving CDM_i in M steps before proceeding to CDM_{i+1}. However, we evolve only *one* FU row in CDM_i before proceeding to CDM_{i+1}. This makes it possible to have a working system in K evolution runs (that is, $1/M$ of the total evolution time). While the recognition accuracy is reduced with only one FU row for each CDM, the system is operational and improves gradually as more FU rows are added for each CDM.

3.3 Fitness Function

A certain set of the available vectors, V_t, are used for training of the system, while the remaining, V_v, are used for verification after the evolution run.

Each row of FUs is fed with the training vectors ($v \in V_t$), and fitness is based on the row's ability to give a positive (1) output for vectors v belonging to its own category ($C_v = C_i$), while giving a negative (0) output for the rest of the vectors ($C_v \neq C_i$).

In the case of a positive output when $C_v = C_i$, the value A is added to the fitness sum. When $C_v \neq C_i$ and the row gives a negative output (value 0), 1 is added to the fitness sum. The other cases do not contribute to the fitness value. The fitness function F for a row can then be expressed in the following way, where o is the output of the FU row:

$$F = \sum_{v \in V_t} x_v \qquad \text{where } x_v = \begin{cases} A \times o \text{ if } C_v = C_i \\ 1 - o \text{ if } C_v \neq C_i \end{cases} \tag{1}$$

For the experiments, a value of $A = 64$ has been used. This emphasis on the positive matches for C_i has shown to speed up the evolution.

3.4 Evolution Parameters

For the evolution, a population size of 30 is used. Elitism is used, thus, the best individual from each generation is carried over to the next generation. The (single point) crossover rate is 0.9, thus the cloning rate is 0.1. A roulette wheel selection scheme is applied. Linear fitness scaling is used, with 6 expected copies of the best individual. The mutation rate is expressed as a probability for a certain number, n, of mutations on each genome. The probabilities are as follows:

n	1	2	3	4
$p(n)$	$\frac{7}{10}$	$\frac{1}{10}$	$\frac{1}{10}$	$\frac{1}{10}$

4 Results

This section presents the results of the experiments undertaken. The results are based on a software simulation of the EHW architecture.

4.1 Architecture Parameters

The architecture parameters N and M, that is, the number of FUs in an FU row and the number of FU rows in a CDM, respectively, have been evaluated. As can be seen in figure 4, the number of generations required to evolve an FU row is dependent of the number of FUs in such a row.

Too few FUs makes it difficult for the FU row to distinguish between the right and wrong categories. Too many FUs give the same problem, as well as having the problem of a longer genome and thus a larger search space. We have not seen

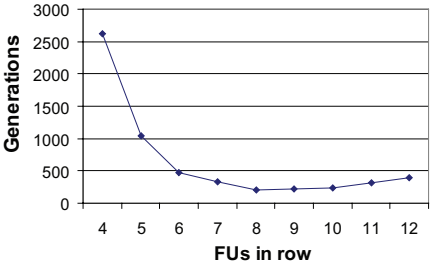


Fig. 4. Generations required to evolve rows of different number of FUs. Average over 5 evolution runs.

any discernable connection between the number of FUs in an FU row and the recognition accuracy, as long as the row could be evolved to a maximum fitness value.

However, increasing the number of FU rows for a category leads to an increase in the recognition accuracy, as seen in figure 5. As the number of FU rows increases, so does the output resolution from each CDM. Each FU row is evolved from an initial random bitstream, which ensures a variation in the evolved FU rows. To draw a parallel to the system in [6], each FU row represents a kernel function. More FU rows give more kernel functions (with different centers) that the unknown pattern can fall into.

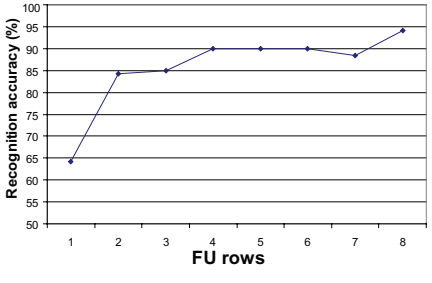


Fig. 5. Accuracy obtained for varying the number of FU rows. A fixed row size of 8 was used. Average over 3 runs.

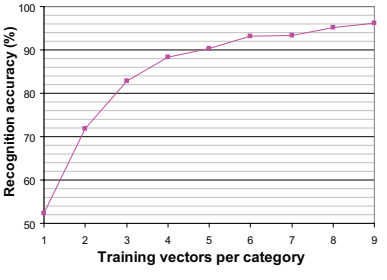


Fig. 6. Accuracy obtained for varying the number of training vectors per category, with $N = 6$ and $M = 10$

4.2 Recognition Accuracy

10 evolution runs were conducted, each with a different selection of test vectors. That is, the 10% of the images which were used as test vectors were chosen differently in each evolution run. For $K = 40$, $M = 8$ and $N = 6$, an average recognition accuracy of 96.25% has been achieved. This result is slightly better than the average accuracy of 94.7% reported from the LoDETT system [6].

An experiment varying the number of training vectors has also been undertaken. 1 to 9 training vectors were used for each category. The rest were used as test vectors for calculating the accuracy. The results can be seen in figure 6. The results are competitive to traditional image recognition algorithms' results on the same dataset, such as Eigenfaces (around 90.0% for 8 training vectors per category) [12], or Fisherfaces (around 95.0% for 8 training vectors) [12], but other methods, such as SVM (98%) [13], perform better.

4.3 Evolution Speed

For $K = 40$, $M = 6$ and $N = 10$, the average number of generations (over 10 runs) required for each evolution run (that is, one FU row) is 219, thus an average of 52560 generations is required for the entire system. The average evolution time for the system is 140s on an Intel Xeon 5160 processor using 1 core. This gives an average of 0.6s for one FU row, or 23.3s for 40 rows (the time before the system is operational). It is expected that a hardware implementation will yield lower training times, as the evaluation time for each individual will be reduced.

4.4 Hardware Implementation

A preliminary implementation of an FU row has been synthesized for an FPGA in order to achieve an impression of resource usage. When synthesized for a Xilinx XC2VP30, 8 FU rows of 6 FUs (for one CDM) uses 328 slices, that is, 2% of the device. In this case, the data selector MUX in the FU is implemented by using time multiplexing, since directly implementing a 64x8-bit multiplexer for each FU requires many resources in the FPGA. The downside of time multiplexing is that each of the 64 pixels must be present on the inputs for a clock cycle before the classification can be made. With an estimate of maximum 10 cycles needed for the bit counter and the maximum detector, the system would require 74 clock cycles in order to classify a pattern. For a 100MHz system this would give more than 1M classifications per second, that is, less than 1μs for one image.

4.5 Discussion

The main improvement of this system over the LoDETT system is the aspect of on-line evolution. This is achieved by adapting a VRC-like architecture, allowing for quick reconfiguration. By selecting a subset of the input pixels for each image, the size of the circuit can be kept down. The drawback of this method is the extra time or resources needed for implementing the pixel selector MUXes. A positive side effect of the on-line adaptation is the increased recognition accuracy.

Real-time adaptation could be achieved by having evolution running on one separate FU row implemented on the same chip as the operational image recognition system. Thus, when there are changes in the training set (e.g. new samples are added), the FU rows can be re-evolved, one at a time, while the main system is operational using the currently best configuration. Since one FU row requires little hardware resources compared to the full system, little overhead is added.

A full system-on-chip hardware implementation is planned. The GA will run on an embedded processor in a Xilinx FPGA. Evolution speed as well as recognition speed should be measured. Furthermore, it would be interesting to improve the architecture in ways of dynamically adjusting the number of FUs and rows for each CDM, depending on its evolvability. Other architectural changes could also be considered, e.g. some kind of hierachical approaches, for increased recognition accuracy or improved resource usage.

Secondly, variations of the architecture should be tested on other pattern recognition problems. Since the main advantage of this architecture is its very high recognition speed, applications requiring high throughput should be suitable. The LoDETT system has been successfully applied to genome informatics and other applications [14,15]. It is expected that the proposed architecture also could perform well on similar problems, if suitable functions for the FUs are found.

5 Conclusions

An EHW architecture for a complex pattern recognition task has been proposed. The architecture supports run-time reconfiguration and is thus suitable for implementation in an on-chip evolution system. The architecture proposed utilizes data buses and higher level functions in order to reduce the search space. In addition, evolution of the system follows an incremental approach. Short evolution time is needed for a basic working system to be operational. Increased generalisation can then be added. The classification accuracy has been shown to slightly better than earlier offline EHW approaches. The system seems suitable for applications requiring high speed and online adaptation to a changing training set.

Acknowledgment

The research is funded by the Research Council of Norway through the project *Biological-Inspired Design of Systems for Complex Real-World Applications* (proj. no. 160308/V30).

References

1. Yao, X., Higuchi, T.: Promises and challenges of evolvable hardware. In Higuchi, T., et al., eds.: Evolvable Systems: From Biology to Hardware. First International Conference, ICES 96. Volume 1259 of Lecture Notes in Computer Science. Springer-Verlag (1997) 55–78
2. Sekanina, L., Ruzicka, R.: Design of the special fast reconfigurable chip using common F PGA. In: Proc. of Design and Diagnostics of Electronic Circuits and Systems - IEEE DDECS'2000. (2000) 161–168

3. Iwata, M., Kajitani, I., Yamada, H., Iba, H., Higuchi, T.: A pattern recognition system using evolvable hardware. In: Proc. of Parallel Problem Solving from Nature IV (PPSN IV). Volume 1141 of Lecture Notes in Computer Science., Springer-Verlag (September 1996) 761–770
4. Torresen, J.: Scalable evolvable hardware applied to road image recognition. In et al., J.L., ed.: Proc. of the 2nd NASA/DoD Workshop on Evolvable Hardware, IEEE Computer Society, Silicon Valley, USA (July 2000) 245–252
5. Torresen, J., Bakke, W.J., Sekanina, L.: Recognizing speed limit sign numbers by evolvable hardware. In: Parallel Problem Solving from Nature. Volume 2004., Springer Verlag (2004) 682–691
6. Yasunaga, M., Nakamura, T., Yoshihara, I., Kim, J.: Genetic algorithm-based design methodology for pattern recognition hardware. In Miller, J., et al., eds.: Evolvable Systems: From Biology to Hardware. Third International Conference, ICES 2000. Volume 1801 of Lecture Notes in Computer Science., Springer-Verlag (2000) 264–273
7. Glette, K., Torresen, J., Yasunaga, M., Yamaguchi, Y.: On-chip evolution using a soft processor core applied to image recognition. In: Proc. of the First NASA /ESA Conference on Adaptive Hardware and Systems (AHS 2006), Los Alamitos, CA, USA, IEEE Computer Society (2006) 373–380
8. Zhao, W., Chellappa, R., Phillips, P.J., Rosenfeld, A.: Face recognition: A literature survey. ACM Comput. Surv. **35**(4) (2003) 399–458
9. Ngo, H., Gottumukkal, R., Asari, V.: A flexible and efficient hardware architecture for real-time face recognition based on eigenface. In: Proc. of IEEE Computer Society Annual Symposium on VLSI, IEEE (2005) 280–281
10. Glette, K., Torresen, J.: A flexible on-chip evolution system implemented on a Xilinx Virtex-II Pro device. In: Evolvable Systems: From Biology to Hardware. Sixth International Conference, ICES 2005. Volume 3637 of Lecture Notes in Computer Science. Springer-Verlag (2005) 66–75
11. Goldberg, D.: Genetic Algorithms in search, optimization, and machine learning. Addison–Wesley (1989)
12. Zhou, D., Yang, X.: Face recognition using enhanced fisher linear discriminant model with facial combined feature. In: PRICAI. Lecture Notes in Computer Science, Springer-Verlag (2004) 769–777
13. Kim, K., Kim, J., Jung, K.: Recognition of facial images using support vector machines. In: Proceedings of the 11th IEEE Signal Processing Workshop on Statistical Signal Processing, IEEE (2001) 468–471
14. Yasunaga, M., et al.: Gene finding using evolvable reasoning hardware. In A. Tyrrel, P.H., Torresen, J., eds.: Evolvable Systems: From Biology to Hardware. Fifth International Conference, ICES'03. Volume 2606 of Lecture Notes in Computer Science. Springer-Verlag (2003) 228–237
15. Yasunaga, M., Kim, J.H., Yoshihara, I.: The application of genetic algorithms to the design of reconfigurable reasoning vlsi chips. In: FPGA '00: Proceedings of the 2000 ACM/SIGDA eighth international symposium on Field programmable gate arrays, New York, NY, USA, ACM Press (2000) 116–125

Learning and Recognition of Hand-Drawn Shapes Using Generative Genetic Programming

Wojciech Jaśkowski, Krzysztof Krawiec, and Bartosz Wieloch

Institute of Computing Science, Poznań University of Technology
Piotrowo 2, 60965 Poznań, Poland
{wjaskowski|kkrawiec|bwieloch}@cs.put.poznan.pl

Abstract. We describe a novel method of evolutionary visual learning that uses generative approach for assessing learner's ability to recognize image contents. Each learner, implemented as a genetic programming individual, processes visual primitives that represent local salient features derived from a raw input raster image. In response to that input, the learner produces partial reproduction of the input image, and is evaluated according to the quality of that reproduction. We present the method in detail and verify it experimentally on the real-world task of recognition of hand-drawn shapes.

1 Introduction

In supervised learning applied to object recognition, the search in the space of hypotheses (learners) is usually guided by some measure of quality of discrimination of training examples from different object classes. This requires defining, somehow arbitrarily, learner's desired response for objects from particular classes (e.g., defining desired combinations of output layer excitations in case of an artificial neural network). Though proven effective in several applications, such an approach suffers from relatively high risk of overfitting, especially when the number of image features is large. For instance, in our past experience with evolutionary synthesis of object recognition systems [1,2], many evolved learners tended to use irrelevant image features, coincidentally correlated with the partitioning of training examples into concepts. This happens because learners are rewarded exclusively for decisions they make, and not for the actual 'understanding' of the recognized objects.

Moreover, applicability of supervised feature-based learning methods is restricted to simple recognition tasks with a limited number of object classes. For more complex objects and for large numbers of object classes, one usually has to rely on model-based approach and explicitly specify the models of objects to be recognized, which is often tedious and time-consuming.

To avoid overfitting on one hand and model-based approach on the other, in this paper we make the learning process unsupervised in the sense that the learner is not explicitly told how it should discriminate the positive examples from the negative ones. Rather than that, it is encouraged to reproduce a selected

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 281–290, 2007.
© Springer-Verlag Berlin Heidelberg 2007

aspect of the object being recognized, which, in turn, enables more thorough evaluation.

In experimental part, we tackle the problem of interpretation of hand-drawn sketches. In real-world scenarios, such recognition systems may be helpful for direct digitization of hand-sketched diagrams, for instance, block diagrams, UML diagrams, etc., saving the time required for tedious manual re-entry of paper notes. Such drawings are typically acquired using input devices like TabletPC computers, PDAs, or graphics tablets (digitizers). Most such devices produce on-line data, i.e., provide both spatial and temporal information about pen (stylus) position. As our approach requires spatial information only, its applicability spans also off-line interpretation of sketches stored as ordinary raster images (e.g., acquired from paper drawing using a scanner).

According to Krishnapuram *et al.* [3], the complete task of sketch interpretation may be subdivided into three subtasks: (i) segmentation of drawing into disjoint shapes that are recognized independently, (ii) fitting of shapes (models) to drawings, and (iii) recognition of particular shapes. The approach described in this paper tackles two latter tasks. However, we discuss the possibility of tackling the segmentation task as well.

The primary contribution of this paper may be summarized as development and practical verification of a novel method to object recognition that (i) uses genetic programming to evolve visual learners, (ii) estimates learner's fitness by assessing its ability to restore essential features of the input image, and (iii) uses visual primitives as basic 'granules' of visual information.

2 Generative Visual Learning

The proposed approach may be shortly characterized as *generative visual learning*, as our evolving learners aim at reproducing the input image and are rewarded according to the quality of that reproduction. The reproduction is partial, i.e., the learner restores only a particular *aspect* of the image contents. In this paper, the aspect of interest is shape, whereas other factors, like color, texture, shading, are ignored.

The reproduction takes place on a virtual canvas spanned over the input image. On that canvas, the agent is allowed to perform some elementary *drawing actions* (DAs for short). To enable successful reproduction, DAs should be compatible with the image aspect that is to be reconstructed. In this paper, we consider hand-drawn polygons and, to enable the learner to restore their shape, we make our DAs draw sections.

As an example, let us consider reconstruction of an empty triangular shape. It requires from the learner performing the following steps: (i) detection of conspicuous features — triangle corners, (ii) pairing of the detected triangle corners, and (iii) performing DAs that connect the paired corners. However, within the proposed approach, the learner is not given *a priori* information about the concept of the corner nor about the expected number of them. We expect the learner to discover these on its own.

To reduce the amount of data that has to be processed and to bias the learning towards the image aspect of interest, our approach abstracts from raster data and relies only on selected salient features in the input image s. For each locally detected feature, we build an independent *visual primitive* (VP for short). The complete set of VPs derived from s is denoted in following by P.

The learning algorithm itself does not make any assumptions about the particular salient feature used for VP creation. Reasonable instances of VPs include, but are not limited to, edge fragments, regions, texems, or blobs. However, the type of detected feature determines the image aspect that is reconstructed. As in this paper we focus on shape, we use VPs representing prominent local luminance gradients derived from s using a straightforward procedure. Each VP is described by three scalars called hereafter *attributes*; these include two spatial coordinates of the edge fragment and the local gradient orientation.

3 GP-Based Learners

On the top level, the proposed method uses evolutionary algorithm that maintains a population of visual learners (individuals, solutions), each of them implemented as genetic programming (GP, [4]) expression. Each visual learner L is a procedure written in a form of a tree, with nodes representing *elementary operators* that process sets of VPs. The terminal nodes (named *ImageNode*s) fetch the set of primitives P derived from the input image s, and the consecutive internal nodes process the primitives, all the way up to the root node. A particular tree node may (i) group primitives, (ii) perform selection of primitives using constraints imposed on VP attributes or their other properties, or (iii) add new attributes to primitives.

Table 1 presents the complete list of GP operators that may reside in particular tree nodes. We use strongly-typed GP (cf. [4]), which implies that two operators may be connected to each other only if their input/output types match. The following types are used: numerical scalars ($\Re$ for short), sets of VPs (Ω, potentially nested), attribute labels (A), binary arithmetic relations (R), and aggregators (G).

The non-terminal GP operators may be divided into following categories:

1) Scalar operators (as in standard GP applied to symbolic regression; see [4]). Scalar operators accept arguments of type $\Re$ and return result of type $\Re$.

2) Selectors. The role of a selector is to filter out some of the VPs it receives from its child node(s) according to some objectives or condition. Selectors accept at least one argument of type Ω and return result of type Ω. *Non-parametric selectors* expect two child nodes of type Ω and produce an output of type Ω. Operators that implement basic set algebra, like set union, intersection, or difference, belong to this category. *Parametric selectors* expect three child nodes of types Ω, A, and $\Re$, respectively, and produce output of type Ω. For instance, operator *LessThan* applied to child nodes (P, p_o, 0.3) filters out all VPs from P for which the value of the attribute p_o (orientation) is less than 0.3.

Table 1. The GP operators

Type	Operator
$\Re$	Ephemeral random constant
Ω	*ImageNode* – the VP representation P of the input image s
A	p_x, p_y, p_o and custom attributes added by *AddAttribute*
R	*Equals*, *Equals5Percent*, *Equals10Percent*, *Equals20Percent*, *LessThan*, *GreaterThan*
G	Sum, Mean, Product, Median, Min, Max, Range
$\Re$	+($\Re$,$\Re$), –($\Re$,$\Re$), *($\Re$,$\Re$), /($\Re$,$\Re$), sin($\Re$), cos($\Re$), abs($\Re$), sqrt($\Re$), sgn($\Re$), ln($\Re$)
Ω	*SetIntersection*(Ω,Ω), *SetUnion*(Ω,Ω), *SetMinus*(Ω,Ω), *SetMinusSym*(Ω,Ω), *SelectorMax*(Ω,A), *SelectorMin*(Ω,A), *SelectorCompare*(Ω,A,R,$\Re$), *CreatePair*(Ω,Ω), *CreatePairD*(Ω,Ω), *ForEach*(Ω,Ω), *ForEachCreatePair*(Ω,Ω,Ω), *ForEachCreatePairD*(Ω,Ω,Ω), *AddAttribute*(Ω,$\Re$), *AddAttributeForEach*(Ω,$\Re$), *GroupHierarchyCount*(Ω,$\Re$), *GroupHierarchyDistance*(Ω, $\Re$), *GroupProximity*(Ω, $\Re$), *GroupOrientationMulti*(Ω, $\Re$), *Ungroup*(Ω), *Draw*(Ω)

3) Iterators. The role of an iterator is to process one by one the VPs it receives from one of its children. For instance, operator *ForEach* iterates over all the VPs from its left child and processes each of them using the GP code specified by its right child. The VPs resulting from all iterations are grouped into one VP and returned.

4) Grouping operators. The role of those operators is to group primitives into a certain number of sets according to some objectives or conditions. For instance, *GroupHierarchyCount* uses agglomerative hierarchical clustering, where euclidean distance of primitives serves as the distance metric.

5) Attribute constructors. An attribute constructor defines and assigns a new attribute to the VP it processes. The definition of a new attribute, which must be based on the values of existing VP attributes, is given by the GP code contained in the right child subtree. To compute the value of a new attribute, attribute constructor passes the VP (operator *AddAttribute*) or the sub-primitives of the VP (operator *AddAtributeToEach*) through that subtree. Attribute constructors accept one argument of type Ω and one of type $\Re$, and return a result of type Ω.

The detailed description of all implemented operators may be found in [5,6].

Given the elementary operators, the learner L applied to an input image s builds gradually a hierarchy of VP sets derived from s. Each application of selector, iterator, or grouping operator creates a new set of VPs that includes other elements of the hierarchy. In the end, the root node returns a nested VP hierarchy built atop of P, which reflects the processing performed by L for s. Some of the elements of the hierarchy may be tagged by new attributes created by attribute constructors.

Figure 1 illustrates an example of VP hierarchy built by the learner in response to input image/stimulus s. In the left part of the figure, the short edge fragments labeled by single lower-case letters represent the original VPs derived from the

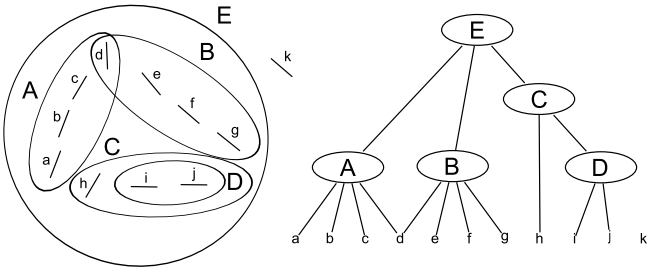


Fig. 1. The primitive hierarchy built by the learner from VPs, imposed on the image (left) and shown in an abstract form (right); VP attributes not shown for clarity

input image s, which together build up P. In the right part, the VP hierarchy is shown in an abstract way, without referring to the actual placement of particular visual primitives in the input image. Note that the hierarchy does not have to contain all VPs from P, and that a particular VP from P may occur in more than one branch of the hierarchy.

Individual's fitness is based on DAs (drawing actions) that it performs in response to visual primitives P derived from training images $s \in S$. To reconstruct the essential features of the input image s, the learner is allowed to perform DAs that boil down to drawing sections on the output canvas c. To implement that within the GP framework, an extra GP operator called *Draw* is included in the set of operators presented in Table 1. It expects as an argument one VP set T and returns it unchanged, drawing on canvas c sections connecting each pair of VPs contained in T.

The drawing created on the canvas c by the learner L for an input image s is then evaluated to provide feedback for L and enable its potential improvement. This evaluation consists in comparing the contents of c to s. For this purpose, a simple and efficient approach was designed. In general, this approach assumes that the difference between c and s is proportional to the minimal total cost of bijective assignment of lit pixels of c to lit pixels of s. The total cost is a sum of costs for each pixel assignment. The cost of assignment depends on the distance between pixels in the following way. When the distance is less than 5, the cost is 0; maximum cost equals 1 when the distance is greater than 15; between 5 and 15 the cost is a linear function of the distance. For pixels that cannot be assigned (e.g., because there are more lit pixels in c than in s), an additional penalty of value 1 is added to the total cost. In order to compute the minimal total cost of assignment a greedy heuristic was applied.

The (minimized) fitness of L is defined as the total cost of the assignment normalized by the number of lit pixels in $s \in S$, averaged over the entire training set of images S. The ideal learner perfectly restores shapes in all training images and its fitness amounts to 0. The more the canvas c produced by a learner differs from s, the greater its fitness value.

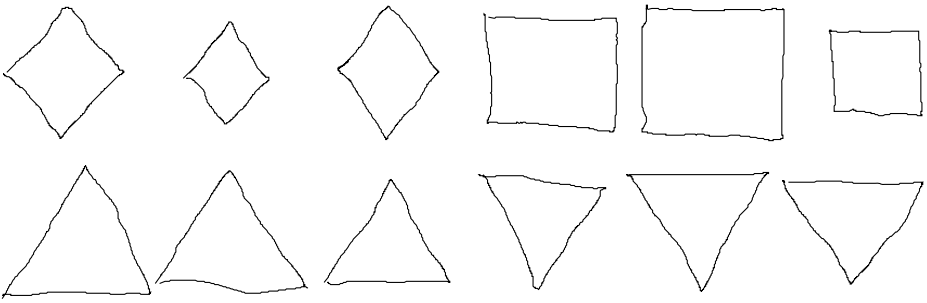


Fig. 2. Selected training examples

4 Related Research in Visual Learning

In most approaches to visual learning reported in literature, learning is limited to parameter optimization that usually concerns only a particular processing step, such as image segmentation, feature extraction, etc. A limited number of learning methods concerns more or less complete recognition systems [7,8,9,2,10,11]. In [1,2] we proposed a methodology that evolved feature extraction procedures encoded either as genetic programming or linear genetic programming individuals. The idea of GP-based processing of attributed visual primitives was explored for the first time in [12], and was further developed in [13,5,6].

The approach presented in this paper may be considered as a variant of generative pattern recognition. In a typical paper on that topic [14], Revow *et al.* used a predefined set of deformable models encoded as B-splines and an elastic matching algorithm based on expectation maximization for the task of handwritten character recognition. In [3], an analogous approach has been proved useful for recognition of hand-drawn shapes. However, the approach presented here goes significantly further, as it does not require *a priori* database of object models. And, last but not least, the recognition (restoration) algorithm has to restore the input image using *multiple* drawing actions, which implies the ability to decompose the analyzed shape into elementary components.

5 Experimental Evaluation

In this section we demonstrate how to apply the approach to recognize and classify hand-drawn sketches (shapes). Using a TabletPC computer we prepared a training set containing 48 images of four elementary shapes: diamonds (D), rectangles (R), triangles pointing upwards (TU), and triangles pointing downwards (TD), each shape represented by 12 examples. The shapes were of different dimensions, orientations and placed at random locations on a raster image of 640×480 pixels. Figure 2 illustrates selected training examples, shown for brevity in one image; note however, that each shape is a separate training example.

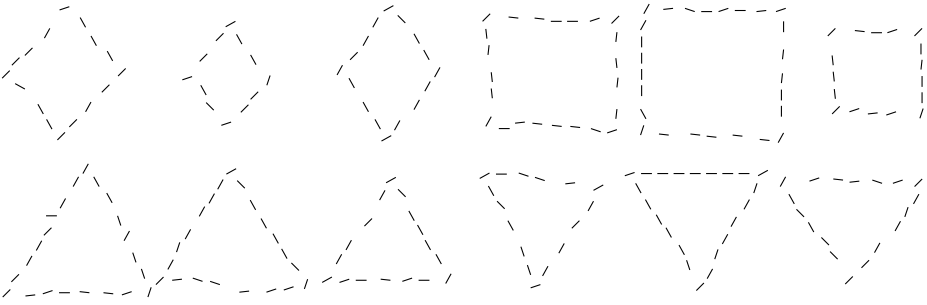


Fig. 3. Visualization of primitives derived from objects depicted in Fig. 2

Figure 3 shows visualization of primitives obtained from objects from Fig. 2. Each segment depicts a single VP, with its spatial coordinates located in the middle of the segment and the orientation depicted by slant.

Technically, we used generative evolutionary algorithm maintaining the population of 25,000 GP individuals for 300 generations. Koza's ramped half-and-half operator with ramp from 2 to 6 [4] was used to produce the initial population. We applied tournament selection with tournament of size 5, using individuals' sizes for tie breaking and thus promoting smaller GP trees. Offspring were created by crossing over selected parent solutions from previous generation (with probability 0.8), or mutating selected solutions (with probability 0.2). The GP tree depth limit was set to 10; the mutation and crossover operations might be repeated up to 5 times if the resulting individuals do not meet this constraint; otherwise, the parent solutions are copied into the subsequent generation. Except for the fitness function implemented for efficiency in C++, the algorithm has been implemented in Java with help of the ECJ package [15]. For evolutionary parameters not mentioned here explicitly, ECJ's defaults have been used.

The experiment was conducted according to the following procedure. First, for each class of shape (D, R, TU and TD), 5 evolutionary runs were performed using the training set for fitness computation. From each run, the best individual (learner) was chosen. These 20 individuals constituted the pool of individuals, which we used to build a recognition system that was subsequently evaluated on a separate test set containing 124 shapes. We present results for two recognition systems:

1) Simple recognition system consists of 4 best-in-class individuals, selected from the pool according to fitness value (based on the training data). This straightforward system performs recognition of the test example t by computing fitnesses of all four individuals for t and indicating the class associated with the fittest individual. The rationale behind such a procedure is following. The learner was taught only to perform well on images from one class and its fitness should be near 0 only for images of this class. For example, it is unlikely that an individual that learned the concept of triangle could recognize squares, thus it will get a high fitness value. Simple recognition system achieves test-set accuracy of classification of 88.71%; the detailed confusion matrix is presented in Table 2a.

Table 2. Test-set confusion matrices for simple recognition system (a) and voting recognition system (b) (rows: actual object classes, columns: system's decisions)

(a)

	D	R	TU	TD
D	**33**	0	3	0
R	1	**25**	3	2
TU	0	0	**28**	0
TD	5	0	0	**24**

(b)

	D	R	TU	TD
D	**34**	1	1	0
R	2	**27**	1	1
TU	0	0	**28**	0
TD	0	0	0	**29**

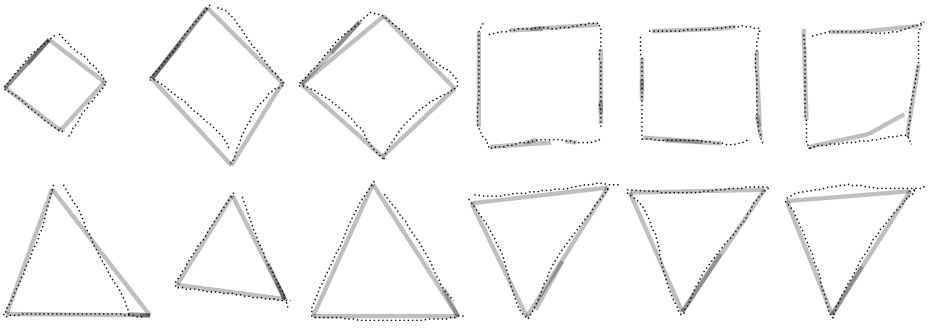


Fig. 4. The generative restoration process performed by trained learners

2) Voting recognizer uses all 20 individuals from the pool and runs multiple voting procedures to utilize the diversity of individuals obtained from different evolutionary runs. Technically, all 5^4 possible combinations of individuals-voters are considered, always using one voter per class. Each voting produces one decision, using the same procedure as the simple recognition system. The class indicated most frequently in all votings is the final decision of the recognition system. This approach performs significantly better than the simple recognition system and attains test-set recognition accuracy of 95.16% (confusion matrix presented in Table 2b).

In Fig. 4, we illustrate the process of generative shape restoration performed by the well-performing individuals for randomly selected test shapes. Thin dotted lines mark the shapes drawn by a human, whereas thick continuous lines depict drawing actions performed by the individual (sections). Darker colors reflect overlapping of multiple DAs. It may be easily observer that, in most cases, the evolved individuals successfully reproduce the overall shape of the recognized object. Reproduction seems to be robust despite various forms of imperfectness of the hand-drawn figures.

In Fig. 5, we present the GP code of selected individual trained to recognize objects from the TU class. It should be emphasized that this individual uses several nodes to issue drawing actions. In this way, the evolved individual exploits the inherent ability of our approach that allows gradual composition of recognized object from primitive components (sections).

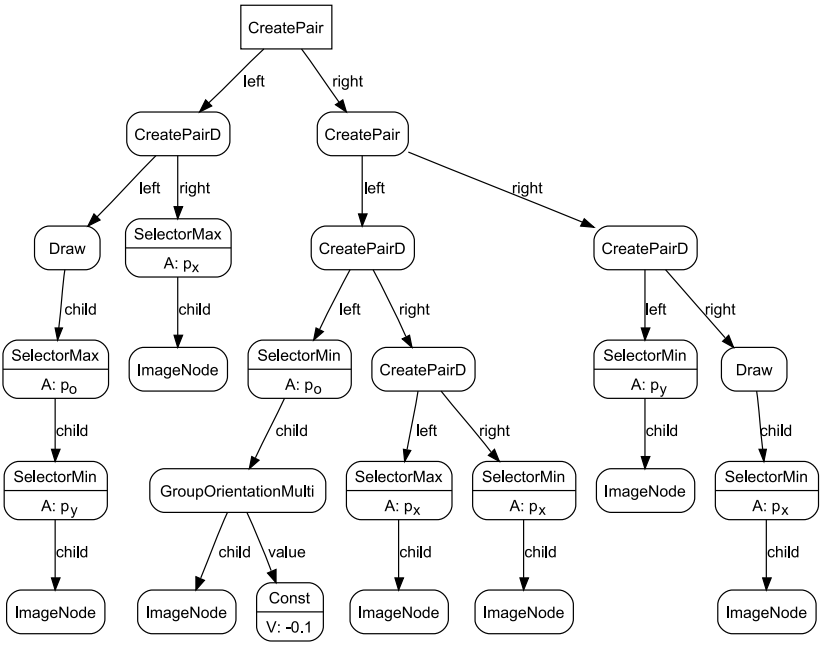


Fig. 5. The GP code of selected well-performing individual trained on TU class

6 Conclusions

The obtained results demonstrate the ability of the approach to evolve generative object recognition systems that successfully classify real-world sketches. This result has been obtained using very limited background knowledge, encoded in GP operators. Learners are not provided with negative examples (each learner is trained only on examples from its own class), so new object classes may be added without the need of modifying the existing recognizers. The method offers low time complexity, resulting mostly from primitive-based processing. Recognition of an object takes on average only 0.1 ms for implementation running on 2.4 GHz AMD Opteron processor.

In this preliminary study we focused on recognition of basic geometrical shapes. Interpretation of more complex images may require more sophisticated and substantially larger GP individuals. To alleviate scalability issues that may arise in such a case, we devised an extended approach that performs automatic decomposition of image processing into multiple GP trees and enables sharing of decomposed trees between multiple learning tasks. Preliminary results indicate that such approach is profitable in terms of convergence speed [5,6].

Future research could concern other aspects of visual information, like color or texture, and other input representation spaces, like region adjacency graphs. It would be also interesting to investigate the possibility of some integration of different aspects of visual stimuli.

References

1. Bhanu, B., Lin, Y., Krawiec, K.: Evolutionary Synthesis of Pattern Recognition Systems. Springer-Verlag, New York (2005)
2. Krawiec, K., Bhanu, B.: Visual learning by coevolutionary feature synthesis. IEEE Transactions on System, Man, and Cybernetics – Part B **35** (2005) 409–425
3. Krishnapuram, B., Bishop, C.M., Szummer, M.: Generative models and bayesian model comparison for shape recognition. In: IWFHR '04: Proceedings of the Ninth International Workshop on Frontiers in Handwriting Recognition (IWFHR'04), Washington, DC, USA, IEEE Computer Society (2004) 20–25
4. Koza, J.: Genetic programming – 2. MIT Press, Cambridge, MA (1994)
5. Jaskowski, W.: Genetic programming with cross-task knowledge sharing for learning of visual concepts. Master's thesis, Poznan University of Technology, Poznań, Poland (2006)
6. Wieloch, B.: Genetic programming with knowledge modularization for learning of visual concepts. Master's thesis, Poznan University of Technology, Poznań, Poland (2006)
7. Teller, A., Veloso, M.: PADO: A new learning architecture for object recognition. In Ikeuchi, K., Veloso, M., eds.: Symbolic Visual Learning. Oxford Press, New York (1997) 77–112
8. Rizki, M., Zmuda, M., Tamburino, L.: Evolving pattern recognition systems. IEEE Transactions on Evolutionary Computation **6** (2002) 594–609
9. Maloof, M., Langley, P., Binford, T., Nevatia, R., Sage, S.: Improved rooftop detection in aerial images with machine learning. Machine Learning **53** (2003) 157–191
10. Olague, G., Puente, C.: The honeybee search algorithm for three-dimensional reconstruction. In Rothlauf, F., Branke, J., Cagnoni, S., Costa, E., Cotta, C., Drechsler, R., Lutton, E., Machado, P., Moore, J.H., Romero, J., Smith, G.D., Squillero, G., Takagi, H., eds.: EvoWorkshops. Volume 3907 of Lecture Notes in Computer Science., Springer (2006) 427–437
11. Howard, D., Roberts, S.C., Ryan, C.: Pragmatic genetic programming strategy for the problem of vehicle detection in airborne reconnaissance. Pattern Recognition Letters **27** (2006) 1275–1288
12. Krawiec, K.: Learning high-level visual concepts using attributed primitives and genetic programming. In Rothlauf, F., ed.: EvoWorkshops 2006. LNCS 3907, Berlin Heidelberg, Springer-Verlag (2006) 515–519
13. Krawiec, K.: Evolutionary learning of primitive-based visual concepts. In: Proc. IEEE Congress on Evolutionary Computation, Sheraton Vancouver Wall Centre Hotel, Vancouver, BC, Canada July 16-21. (2006) 4451–4458
14. Revow, M., Williams, C.K.I., Hinton, G.E.: Using generative models for handwritten digit recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence **18** (1996) 592–606
15. Luke, S.: ECJ evolutionary computation system (2002) (http://cs.gmu.edu/eclab/projects/ecj/).

Multiclass Object Recognition Based on Texture Linear Genetic Programming

Gustavo Olague[1], Eva Romero[1], Leonardo Trujillo[1], and Bir Bhanu[2]

[1] CICESE, Km. 107 carretera Tijuana-Ensenada, Mexico
olague@cicese.mx
http://cienciascomp.cicese.mx/evovision/
[2] Center for Research in Intelligent Systems,
University of California, Riverside, USA

Abstract. This paper presents a linear genetic programming approach, that solves simultaneously the region selection and feature extraction tasks, that are applicable to common image recognition problems. The method searches for optimal regions of interest, using texture information as its feature space and classification accuracy as the fitness function. Texture is analyzed based on the gray level cooccurrence matrix and classification is carried out with a SVM committee. Results show effective performance compared with previous results using a standard image database.

1 Introduction

Recognition is a classical problem in computer vision whose task is that of determining whether or not the image data contains some specific object, feature, or activity. This task can normally be solved robustly by a human, but is still not satisfactory solved by a computer for the general case: arbitrary objects in arbitrary situations. Genetic and evolutionary algorithms have been used to solve recognition problems in recent years. Tackett [1] presented one of the first works that applied genetic and evolutionary algorithms to solve recognition problems. The author used genetic programming (GP) to assign detected image features to classify vehicles such as tanks on US ARMY NVEOD terrain board imagery. In this work the genetic programming approach outperformed a neural network, as well as binary tree classifier on the same data, producing lower false positives. Teller and Veloso [2,3] apply also genetic programming to perform face recognition tasks based on the PADO language using a local indexed memory. The method was tested on a classification of 5 classes and achieved 60% of accuracy for images without noise. Howard et al. [4,5] propose a multi-stage genetic programming approach to evolve fast and accurate detectors in short evolution times. In a first stage the GP takes a random selection of non-object pixels and all the object pixels from the ground truth as test points. The fittest detector from the evolution is applied in order to produce a set of false positives (FP). A second stage of GP uses the discovered FP and all of the object pixels from the truth as test points to evolve a second detector. Then, the fittest detectors

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 291–300, 2007.
© Springer-Verlag Berlin Heidelberg 2007

from both stages are combined and in order to detect objects having large variability this two-stage GP process is further extended into a number of stages. This multi-stage method stops when enough sub-detectors exist to detect all objects. Zhang et al. [6] use GP for domain independent object detection problems in which the locations of small objects of multiple classes in large images must be found. They consider three terminal sets based on domain independent pixel statistics and consider also two different function sets. The fitness function is based on the detection rate and the false alarm rate. The approach was tested on three object detection problems where the objects are approximately the same size and the background is not too cluttered. Lin and Bhanu [7] propose a co-evolutionary genetic programming (CGP) approach to learn composite features for object recognition. The motivation of using genetic programming is to overcome the limitations of human experts who consider only a small number of conventional combinations of primitive features. In this way, their CGP method can try a very large number of unconventional combinations which may yield exceptionally good results. Experiments with SAR images show that CGP could learn good composite features in order to distinguish from several classes. Krawiec and Bhanu [8] propose to use linear genetic programming to represent feature extraction agents within a framework of cooperative coevolution in order to learn feature-based recognition tasks. Experiments on demanding real-world tasks of object recognition in synthetic aperture radar imagery, shows the competitiveness of the proposed approach with human-designed recognition systems. Roberts and Claridge [9] present a system whereby a feature construction stage is simultaneously coevolved along side the GP object detectors. In this way, the proposed system is able to learn both stages of the visual process simultaneously. Initial results in artificial and natural images show how it can quickly adapt to form general solutions to difficult scale and rotation invariant problems.

This work proposes a general multiclass object recognition system to be tested in a challenging image database commonly used in computer vision research [10]. Categorization is the name in computer vision for the automatic recognition of object classes from images. This task is normally posed as a learning problem in which several classes are partitioned into sets for training and testing. The goal is to show that high classification accuracy is feasible for three object classes on photographs of real objects viewed under general lighting conditions, poses and viewpoints. The set of test images used for validation comprise photographs obtained from a standard image database, as well as images from the web in order to show the generality of the proposed approach. The proposed method performs well on texture-rich objects and structure-rich ones, because is based on the cooccurrence matrix. We decide to represent the feature extraction procedure with individuals following the linear genetic programming technique, a hybrid of genetic algorithms and genetic programming, which has the advantage of being able to control the elements in the tree structure. In this way, each element of the tree structure is evolved only with the respective elements of other elements in the population. This characteristic gives the particularity of being positional

allowing the emergence of substructures and avoiding the destructive effect of crossover, which is considered as a mere mutation in regular GP [11,12,8].

2 Texture Analysis and the Gray Level Cooccurrence Matrix

Image texture analysis has been a major research area in the field of computer vision since the 1970's. Historically, the most commonly used methods for describing texture information are the statistical based approaches. First order statistical methods use the probability distribution of image intensities approximated by the image histogram. With such statistics, it is possible to extract descriptors that characterize image information. First order statistics descriptors include: entropy, kurtosis and energy, to name but a few. Second order statistical methods represent the joint probability density of the intensity values (gray levels) between two pixels separated by a given vector $\boldsymbol{V}$. This information is coded using the *Gray Level Cooccurrence Matrix* (GLCM) $M\left(i,j\right)$ [13,14]. Statistical information derived from the GLCM has shown reliable performance in tasks such as image classification [15] and content based image retrieval [16,17].

Formally, the GLCM $M_{i,j}(\pi)$ defines a joint probability density function $f(i,j|\boldsymbol{V},\pi)$ where i and j are the gray levels of two pixels separated by a vector $\boldsymbol{V}$, and $\pi = \{\boldsymbol{V}, R\}$ is the parameter set for $M_{i,j}(\pi)$. The GLCM identifies how often pixels that define a vector $\boldsymbol{V}\left(d,\theta\right)$, and differ by a certain amount of intensity value $\Delta = i - j$ appear in a region R of a given image I. Where $\boldsymbol{V}$ defines the distance d and orientation θ between the two pixels. The direction of $\boldsymbol{V}$, can or cannot be taken into account when computing the GLCM.

The GLCM presents a problem when the number of different gray levels in region R increase, turning difficult to handle or use directly due to the dimensions of the GLCM. Fortunately, the information encoded in the GLCM can be expressed by a varied set of statistically relevant numerical descriptors. This reduces the dimensionality of the information that is extracted from the image using the GLCM. Extracting each descriptor from an image effectively maps the intensity values of each pixel to a new dimension. In this work, the set Ψ of descriptors [14] extracted from $M\left(i,j\right)$ is formed by the following: Entropy, Contrast, Homogeneity, Local homogeneity, Directivity, Uniformity, Moments, Inverse moments, Maximum probability, and Correlation.

3 Evolutionary Learning of Texture Features

The general methodology that is proposed here considers the identification of Regions of Interests (ROIs) and the selection of the set of features of interest (texture descriptors). Thus, visual learning is approached with an evolutionary algorithm that searches for optimal solutions to the multiclass object recognition problem. Two tasks are solved simultaneously. The first task consists in identifying a set of suitable regions where feature extraction is to be performed.

The second task consists in selecting the parameters that define the GLCM, as well as the set of descriptors that should be computed. The output of these two tasks is taken as input by a SVM committee that gives the experimental accuracy on a multiclass problem for the selected features and ROIs.

Linear Genetic Programming for Visual Learning. The learning approach accomplishes a combined search and optimization procedure in a single step. The LGP searches for the best set Ω of ROIs for all images and optimizes the feature extraction procedure by tuning the GLCM parameter set $\pi_i \ \forall \omega_i \in \Omega$ through the selection of the best subset $\{\beta_1 ... \beta_m\}$ of mean descriptor values from the set of all possible descriptors Ψ , to form a feature vector $\boldsymbol{\gamma}_i = (\beta_1 ... \beta_m)$ for each $\omega_i \in \Omega$. Using this representation, we are tightly coupling the ROI selection step with the feature extraction process. In this way, the LGP is learning the best overall structure for the recognition system in a single closed loop learning scheme. Our approach eliminates the need of a human designer, which normally combines the ROI selection and feature extraction steps. Now this step is left up to the learning mechanism. Each possible solution is coded into a single binary string. Its graphical representation is depicted in figure 1. The entire chromosome consists of a tree structure of r binary and real coded strings, and each set of variables is evolved within their corresponding group. The chromosome can be better understood by logically dividing it in two main sections. The first one encodes variables for searching the ROIs on the image, and the second is concerned with setting the GLCM parameters and choosing appropriate descriptors for each ROI.

ROI Selection. The first part of the chromosome encodes ROI selection. The LGP has a hierarchical structure that includes both control and parametric variables. The section of structural or control genes c_i determine the state (on/off)

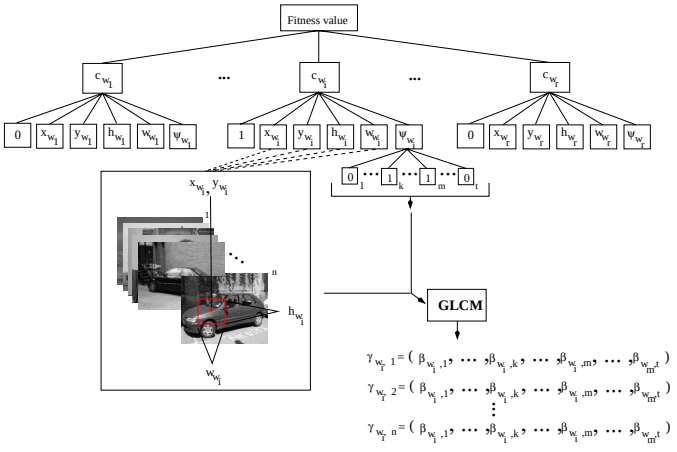


Fig. 1. LGP uses a tree structure similar to the Multicellular Genetic Algorithm [12]

of the corresponding ROI definition blocks ω_i. Each structural gene activates or deactivates one ROI in the image. Each ω_i establishes the position, size and dimensions of the corresponding ROI. Each ROI is defined with four degrees of freedom around a rectangular region: height, width, and two coordinates indicating the central pixel. The choice of rectangular regions is not related in any way with our visual learning algorithm. It is possible to use other types of regions; e.g., elliptical regions, and keep the same overall structure of the LGP. The complete structure of this part of the chromosome is coded as follows:

1. r structural variables $\{c_1....c_r\}$, represented by a single bit each. Each one controls the activation of one ROI definition block. These variables control which ROI will be used in the feature extraction process.
2. r ROI definition blocks $\omega_1....\omega_r$. Each block ω_i, contains four parametric variables $\omega_i = \{x_{\omega_i}, y_{\omega_i}, h_{\omega_i}, w_{\omega_i}\}$, where the variables define the ROIs center $(x_{\omega_i}, y_{\omega_i})$, height (h_{ω_i}) and width (w_{ω_i}). In essence each ω_i establishes the position and dimension for a particular ROI.

Feature Extraction. The second part of the solution representation encodes the feature extraction variables for the visual learning algorithm. The first group is defined by the parameter set π_i of the GLCM computed at each image ROI $\omega_i \in \Omega$. The second group is defined as a string of eleven decision variables that activate or deactivate the use of a particular descriptor $\beta_j \in \Psi$ for each ROI. Since each of these parametric variables are associated to a particular ROI, they are also dependent on the state of the structural variables c_i. They only enter into effect when their corresponding ROI is active (set to 1). The complete structure of this part of the chromosome is as follows:

1. A parameter set π_{ω_i} is coded $\forall\omega_i \in \Omega$, using three parametric variables. Each $\pi_{\omega_i} = \{R_{\omega_i}, d_{\omega_i}, \theta_{\omega_i}\}$ describes the size of the region R, distance d and direction θ parameters of the GLCM computed at each ω_i. Note that R is a GLCM parameter, not to be confused with the ROI definition block ω_i.
2. Eleven decision variables coded using a single bit to activate or deactivate a descriptor $\beta_{j,\omega_i} \in \Psi$ at a given ROI. These decision variables determine the size of the feature vector $\boldsymbol{\gamma}_i$, extracted at each ROI in order to search for the best combination of GLCM descriptors. In this representation, each β_{j,ω_i} represents the mean value of the jth descriptor computed at ROI ω_i.

Classification and Fitness Evaluation. Since the recognition problem aims to classify every extracted region ω_i, we implement a SVM committee that uses a voting scheme for classification. The SVM committee Φ, is formed by the set of all trained SVMs $\{\phi_i\}$, one for each ω_i. The compound feature set $\Gamma = \{\boldsymbol{\gamma}_{\omega i}\}$ is fed to the SVM committee Φ, where each $\boldsymbol{\gamma}_{\omega i}$ is the input to a corresponding ϕ_i. The SVM Committee uses voting to determine the class of the corresponding image. In this way, the fitness function is computed with the *Accuracy*, which is the average accuracy of all SVMs in Φ for a given individual. In other words, $Accuracy = \frac{1}{|\Phi|}\sum_x Acc_{\phi_x}$, summed $\forall\phi_x \in \Phi$, where Acc_{ϕ_x} is the accuracy of the ϕ_j SVM.

SVM Training Parameters. SVM implementation was done using libSVM [18], a C++ open source library. For every $\phi \in \Phi$, the parameter setting is the same for all the population. The SVM parameters are:

- *Kernel Type:* A *Radial Basis Function* (RBF) kernel was used, given by:

$$k(x, x_i) = exp(-\frac{\|x - x_i\|^2}{2\sigma^2}) \tag{1}$$

  The RBF shows a greater performance rate for classifying non linear problems than other types of kernels.
- *Training Set:* The training set used was extracted from 90 different images.
- *Cross Validation:* In order to compute the accuracy of each SVM, we perform k-fold cross validation, with k=6. In general, the accuracy computed with crossvalidation will out perform any other type of validation approach [19]. In k-fold cross validation the data is divided into k subsets of (approximately) equal size. The SVM was trained k times, each time leaving out one of the subsets from training, but using only the omitted subset to compute the classifiers accuracy. This process is repeated until all subsets have been used for both testing and training and the computed average accuracy was used as the performance measure for the SVM.

4 Experiments with the CALTECH Image Database

The image database [10] contains 240 images from which 120 images contain several objects and the other 120 correspond to the same images that have been segmented manually. These images contain objects with different lighting conditions, in different positions, and with several viewpoints. Each image is recorded in RGB format with a size of 320 × 213 pixels. The objects belong to 7 classes: building, trees, cows, airplanes, faces, cars, and bicycles. Because the number of images was insufficient we add more images from the web. We select three classes to test the proposed system: *building*, *faces*, and *cars*. We have two categories for each class: one set of 30 images for training (from [10], see Figures 2(a), 3(a) y 4(a)) and one set of 50 images for testing (from the web, see Figures 2(b), 3(b) y 4(b)). All images were cropped to gray level with a size of 128 × 128 pixels. The parameters of the LGP were 85% crossover, 15% mutation, 80 generations, and 80 individuals. Next, two noteworthy individual solutions are presented:

Individual 92.22%. This individual performs very well with a high average accuracy for training that achieves 92.22%, while the testing is quite good with 73%. This difference is due to the new characteristics of the images downloaded from the web. The LGP selects only one big region located in the lower part of the image because most of the cars are in this part of the images. The best individual obtained in this case is depicted in Figure 5(a), and a photograph with the corresponding ROI is shown in Figure 5(b). Table 4 presents the confusion matrix for this individual applied on the "testing databases".



(a) Set of training images (b) Set of testing images

Fig. 2. Images for the class "building"

Table 1. Confusion matrix obtained for the testing set: 73%

	Building	*Faces*	*Cars*
Building	**68%**	20%	12%
Faces	18%	**78%**	4%
Cars	14%	14%	**72%**



(a) Set of training images (b) Set of testing images

Fig. 3. Images for the class "faces"

Table 2. Confusion matrix obtained for the testing set: 80%

	Building	*Faces*	*Cars*
Building	**85%**	11%	4%
Faces	6%	**80%**	14%
Cars	12%	12%	**76%**



(a) Set of training images (b) Set of testing images

Fig. 4. Images for the class "cars"

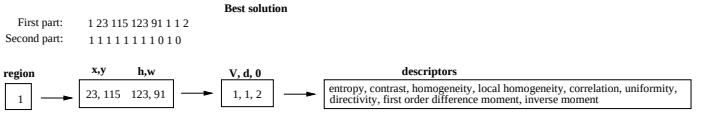
Table 3. Comparison of the Recognition Accuracy

	NN $k = 2000$	NN $k = 216$	Gaussian	LGP
Feature Selection	Hand	Hand	Hand	Automatic
Accuracy	76.3%	78.5%	77.4%	80.0%

Individual 88.88%. Another solution corresponding to an individual with an average accuracy for training of 88.88% was selected to show the level of classification. Its average during testing was as high as 80% because the set of testing images is composed only by the more similar images with respect to the training stage. Similar to the previous case the best individual selects the lower part of the images due to its characteristics. This individual is depicted in Figure 5(c), and a photograph with the corresponding ROI is shown in Figure 5(d). Table 4 presents the confusion matrix for this individual applied on the "testing databases".

Comparison with Other Approaches. The advantage of using a standard database is that it is possible to compare with previous results. For example, in [20] the authors proposed a method that classifies a region according to the proportion of several visual words. The visual words and the proportion of each object are learned from a set of training images segmented by hand. Two methods were used to evaluate the classification: nearest neighbor and Gaussian model. On the average [20] achieved 93% of classification accuracy using the segmented images; while on average the same method achieves 76% choosing the regions by hand. This last result is comparable to our result. Several aspects could be mentioned:

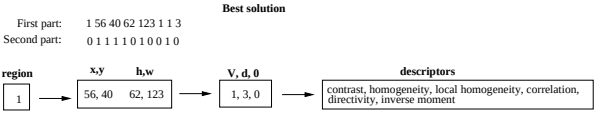
- The approach proposed in this paper does not use segmented images.
- The ROI was automatically selected by the LGP.
- The images used in the testing stage does not belong to the original database [10], these images with a bigger difference were obtained from the web.



(a) Best individual with an Accuracy of 92.22%



(b) ROI found by the LGP system



(c) Best individual with an Accuracy of 88.88%



(d) ROI found by the LGP system

Fig. 5. This images show the best individuals that were found by the LGP approach

We could say that the system presents a positive comparison with respect to the work published in [20]. Table 3 shows the comparison of our approach against those proposed by [20].

5 Conclusions

This paper has presented a general approach based on linear genetic programming to solve multiclass object recognition problems. The proposed strategy searches simultaneously the optimal regions and features that better classify three different object classes. The results presented here show effective performance compared with state-of-the-art results published in computer vision literature.

Acknowledgments. This research was funded by a UC MEXUS-CONACyT Collaborative Research Grant 2005 through the project "Intelligent Robots for the Exploration of Dynamic Environments". This research was also supported by the LAFMI project. Second and third authors supported by scholarships 188966 and 174785 from CONACyT. First author gratefully acknowledges the support of Junta de Extremadura granted when Dr. Olague was in sabbatical leave at the Universidad de Extremadura in Merida, Spain.

References

1. Tackett, W. A.: Genetic programming for feature discovery and image discrimination. In Stephanie Forrest Editor, Proceedings of the 5th International Conference on Genetic Algorithms. University of Illinois at Urbana-Champaign (1993) 303–309
2. Teller, A., Veloso, M.: A controlled experiment: Evolution for learning difficult image classification. In proceedings of the 7th Portuguese Conference on Artificial Intelligence. **LNAI 990** (1995) 165–176

3. Teller, A., Veloso, M.: PADO: Learning tree structured algorithms for orchestration into an object recognition system. Tech. Rep. CMU-CS-95-101, Department of Computer Science, Carnegie Mellon University, Pittsburgh, Pa, USA, (1995)
4. Howard, D., Roberts, S. C., Brankin, R.: Target detection in SAR imagery by genetic programming. Advances in Engineering Software. **30** (1993) 303–311
5. Howard, D., Roberts, S. C., Ryan, C.: The boru data crawler for object detection tasks in machine vision. Applications of Evolutionaty Computing, Proceedings of Evo Workshops 2002. **LNCS 2279** (2002) 220–230
6. Zhang, M., Ciesielski, V., Andreae, P.: A domain independent window-approach to multiclass object detection using genetic programming. EURASIP Journal on Signal Processing, Special Issue on Genetic and Evolutionary Computation for Signal Processing and Image Analysis. **8** (2003) 841–859
7. Lin, Y., Bhanu, B.: Learning features for object recognition. In proceedings of Genetic and Evolutionary Computation. **LNCS 2724** (2003) 2227–2239
8. Krawiec, K., Bhanu, B.: Coevolution and Linear Genetic Programming for Visual Learning. In proceedings of Genetic and Evolutionary Computation. **LNCS 2723** (2003) 332–343
9. Roberts, M. E., Claridge, E.: Cooperative coevolution of image feature construction and object detection. Parallel Problem Solving from Nature. **LNCS 3242** (2004) 899–908
10. CALTECH, "Caltech categories," `http://www.vision.caltech.edu/html-files/archive.html`, 2005.
11. Banzhaf, W., Nordic, P., Keller, R., Francine, F.: Genetic programming: An introduction: On the automatic evolution of computer programs and its applications. San Francisco, CA: Morgan Kaufmann, (1998)
12. Olague, G., Mohr, R.: Optimal camera placement for accurate reconstruction. Pattern Recognition, **34(5)** (2002) 927–944
13. Haralick, R. M., Shanmugam, K., Dinstein, I.: Textural features for image classification. IEEE Trans. on Systems, Man and Cybernetics, **3(6)** (1973) 610-621
14. Haralick, R. M.: Statistical and structural approaches to texture. In proceedings of the IEEE, **67(5)** (1979) 786–804
15. Kjell, J.: Comparative study of noise-tolerant texture classification. IEEE Int. Conference on Systems, Man, and Cybernetics. 'Humans, Information and Technology', **3** (1994) 2431–2436
16. Howarth, P., Rüger, S. M.: Evaluation of texture features for content-based image retrieval. Third International Conference on Image and Video Retrieval, LNCS 3115 (2004) 326–334
17. Ohanian, P. P., and Dubes, R. C.: Performance evaluation for four classes of textural features. Pattern Recognition, **25(8)** (1992) 819–833
18. Chih-Chung Chang and Chih-Jen Lin. *LIBSVM: a library for support vector machines*, 2001.
19. Goutte, C.: Note on free lunches and cross-validation. Neural Computation, **9(6)** (1997) 1245–1249
20. Winn, J., Criminisi, A., Minka, T.: Object categorization by learned universal visual dictionary. 10th IEEE International Conference on Computer Vision. **2** (2005) 1800–1807

Evolutionary Brain Computer Interfaces

Riccardo Poli[1], Caterina Cinel[2], Luca Citi[1,3], and Francisco Sepulveda[1]

[1] Department of Computer Science, University of Essex, UK
[2] Department of Psychology, University of Essex, UK
[3] IMT Institute for Advanced Studies, Lucca, Italy

Abstract. We propose a BCI mouse and speller based on the manipulation of P300 waves in EEG signals. The 2–D motion of the pointer on the screen is controlled by directly combining the amplitudes of the output produced by a filter in the presence of different stimuli. This filter and the features to be combined within it are optimised by a GA.

1 Introduction

Brain-Computer Interfaces (BCIs) measure signals (often EEG) of brain activity intentionally and unintentionally induced by the user and translate them into device control signals (see [10] for a comprehensive review). BCI studies have showed that non-muscular communication and control is possible and might serve useful purposes for those who cannot use conventional technologies.

Event related potentials (ERPs) are among the signals used in BCI studies to date. ERPs are relatively well defined shape-wise variations to the ongoing EEG elicited by a stimulus and temporally linked to it. ERPs include an exogenous response, due to the primary processing of the stimulus, as well as an endogenous response, which is a reflection of higher cognitive processing induced by the stimulus [5,8]. The P300 wave is a large positive ERP with a latency of about 300 ms which is elicited by rare and/or significant stimuli. P300 potentials vary according to the degree of attention the user dedicates to a stimulus. It is, therefore, possible to use them in BCI systems to determine user intentions.

One important application of BCI is controlling 2–D pointer movements. There have been some attempts to develop BCI systems for this purpose, the most successful of which being based on frequency analysis (e.g., [11]), and those using invasive cortical interfaces (e.g., [6]). The former, however, require lengthy training periods before users can control them, while the latter are not very practical requiring surgery, presenting risks of infections, etc. These problems could be overcome by non-invasive systems based on the use of P300s. To date, however, only very limited studies with this approach have been performed [9,1] and these have reported promising but limited successes.

We use an evolutionary approach to parameter identification and optimisation in a P300-based BCI system for the two-dimensional control of a cursor and spelling on a computer. Because of their effectiveness, EAs are frequently applied in the area of image and signal processing. However, no application of EC in

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 301–310, 2007.
© Springer-Verlag Berlin Heidelberg 2007

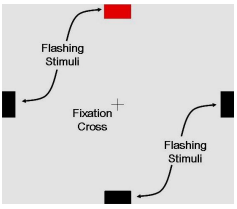


Fig. 1. Our BCI mouse when the stimulus "up" is presented

the area of BCI has been reported with the exception of our own preliminary results [4] where we used an evolutionary approach to process speller data.

The paper is organised as follows. In Sect. 2 we provide details on the participants, the stimuli, and the protocol followed during the training and use of our BCI system. We describe the various phases through which EEG signals are translated into control commands in the system in Sect. 3. In Sect. 4 we provide the details of the GA used. We report on the experimentation of the system in Sect. 5. Finally, we draw some conclusions in Sect. 6.

2 Experimental Methods

In building our BCI system we drew inspiration from Donchin Speller [8], where P300s were used to determine on which stimulus, out of a large set of stimuli concurrently presented, an observer's attention was focused. In particular, we used the following paradigm. Four gray rectangles are constantly superimposed on whatever is shown on a computer screen. They are unobtrusive being relatively small and aligned with the upper, lower, left and right borders of the screen (see Figure 1). Each rectangle corresponds to a possible direction of movement for the mouse cursor.

At 180 ms intervals, this static display is altered by temporarily changing the colour of one of the rectangles from gray to bright red. The stimulus remains brightly coloured for 100 ms, after which it becomes gray again. Which particular rectangle is selected for flashing is determined randomly without replacement. After each flash, the display returns to its standard gray colouring for 80 ms. To limit perceptual errors that can reduce the accuracy of BCI systems [2], the last rectangle to flash in each series is not allowed to be the first to flash in the following series.

The system has three modes of operation: training, validation and normal use. During the phases of acquisition of a training set for our control system, the experimenter selected one of the rectangles on the screen as a target, and participants were asked to focus their attention only on the target stimulus. During training, the screen had a light gray background with no stimuli other than the rectangles mentioned above. During validation phases, participants were asked to perform the same task with the same stimuli and an homogeneous background, except that, at this stage, the system had already been optimised

and we could immediately show participant the trajectory of the mouse pointer produced by their efforts. During normal use, of course, all sorts of windows and icons were present on the screen in addition to the rectangles necessary to control the BCI mouse.

During training and validation, we used the sequences of stimuli described above. Six participants were tested (aged between 23 and 44) of which one has a neuromuscular disability. Each run of our experiment involved presenting a full series of 4 flashing rectangles for 30 times. Participants were requested to count the flashes of one of the rectangles representing direction of motion. The process was repeated for each of the four possible directions, multiple times for each direction. Every few runs, participants were given a few minutes to rest.

We used a 19 channel setup in a Mindset24 System and an electrode cap compliant with the 10-20 international standard to acquire and digitise EEG signals. Efforts were made to obtain impedances below 7 kΩ in all experiments.

3 The System

Each EEG channel is lowpass filtered using a FIR filter of order $N = 30$. The coefficients of the filter were obtained via a least mean squares fit between the frequency response of a target ideal filter and the FIR filter's frequency response. The sampling frequency was 256 Hz, the frequency below which the signal was to be left unaltered was $f_{pass} = 34$ Hz, while the frequency above which all components of the signal were required to be totally suppressed was $f_{stop} = 47$ Hz. After low pass filtering, the signal is decimated to 128 Hz. To extract signal features, we then performed a Continuous Wavelet Transform (CWT) on the 19 channels.

The features are extracted for each epoch of the signal. An *epoch* is a window of 1 second starting when a stimulus is presented on the screen. In each epoch the system needs to process the EEG signals and appropriately emphasise and utilise a P300 if this is present in the epoch.

In our system, pointer control is determined by a filter which is applied to each epoch. In principle, the filter can use all of the coefficients of the CWT of an epoch. We use 30 different unequally-spaced scales between 2 and 40. Since P300s occur within a well known time window after a stimulus, we compute CWT only for 40 samples corresponding to a temporal window from 235 ms and 540 ms after the beginning of each epoch. This gives us a 3–D array $\mathbf{V}(c, s, t)$ of features, where c indexes the channel, s the scale and t the time corresponding to a feature. In total we have $22,800$ components.

Our controller performs a linear combination of a subset of elements of the feature matrix $\mathbf{V}$:

$$P(\mathbf{V}) = a_0 + \sum_{j=1}^{N} a_j \cdot \mathbf{V}(c_j, s_j, t_j). \tag{1}$$

where N is the number of terms in the filter, the coefficients c_j, s_j, t_j identify which component of $\mathbf{V}$ is used as the in the j-th term, and finally the values a_j are coefficients weighing the relative effect of each term.

The value $P(\mathbf{V})$ is then passed through a squashing function ϕ to produce a classifier output, $O(\mathbf{V}) = \phi(P(\mathbf{V}))$, in some interval $[-\Phi, \Phi]$, where Φ is a positive constant. The value of $O(\mathbf{V})$ is an indication of the degree to which an epoch contains a *target*, i.e., the stimulus on which a participant is focusing his/her attention for the purpose of selecting a particular direction of movement for the mouse cursor.

Every time a stimulus flashes, an epoch starts. For each epoch the system records the position of the corresponding stimulus on the screen and acquires and processes a 1 second segment of EEG signal. Epochs acquired during the period of training are annotated also with the direction the participant was asked to focus on.

In epochs corresponding to target stimuli, we expect to find a P300 wave, while in epochs where a non-target stimulus flashed this should not be present.

The motion of the pointer is directly determined by the squashed output of the filter. More precisely, the vertical motion of the pointer is proportional to the difference between the output produced by the filter when processing an epoch where the "up" rectangle was flashed and the output produced by the filter when processing an epoch where the "down" rectangle was flashed. Similarly, the horizontal motion of the pointer is determined by the difference between the outputs produced in response to the flashing of the "right" and "left" rectangles.

In order to turn P300s into mouse pointer motion, we divide the stream of epochs into groups of four. Each group contains epochs corresponding to the flashing of all four possible stimuli. As soon as a full group is acquired, the features and the output of the function $P(\mathbf{V})$ are computed for each of the four epochs. As a squashing function ϕ we used arctan.

As a result of these operations we obtain a tuple of output values (O_u, O_d, O_l, O_r), where subscripts refer to the up, down, left and right stimuli, respectively. These are used to compute the motion vector $v = (v_x, v_y)$ for the mouse cursor on the screen, as follows: $v_x = O_r - O_l$ and $v_y = O_u - O_d$.

4 Evolutionary Algorithm

Given the high noise and limited information content in EEG signals and the difficulty of the task of recognising P300s, it appeared immediately obvious that in our BCI mouse we had to search concurrently for the best features and the best parameters to use for control. It was also obvious that the size of the search space and its discontinuities required a robust search algorithm such as a GA. In our system the GA needs to choose the best features and classifier parameters to control the 2–D motion of the pointer.

For each participant, the GA has the task of identifying a high-quality set of parameters for Equation 1. These include: $N + 1$ real-valued coefficients a_j, N feature channels c_j, which are integers in the set $\{0, \cdots, 18\}$, N integer feature scales s_j in the set $\{0, \cdots, 29\}$, and N integer feature samples t_j in the set $\{0, \cdots, 39\}$.

The operation of feature selection is performed by the GA choosing N tuples (c_j, s_j, t_j), while the classifier training is performed by optimising the corresponding coefficients a_j. These operations are performed jointly by the GA.

The representation used here is simply a concatenation of the floating-point and integer parameters of the linear classifier in Equation 1. For simplicity, we decided to encode integer parameters c_j, s_j and s_j as real numbers, taking care, of course, of rounding them to the nearest integer before using them in Equation 1 (e.g., during fitness evaluation). We were then able to use any operators available for real-coded GAs. In particular, we decided to use blend crossover [7], also known as BLX-α, where the offspring $(a_1^s, \ldots, a_i^s, \ldots, a_n^s)$ is obtained, component by component, as $a_i^s = a_i^1 + c_i(a_i^2 - a_i^1)$, where, for each i, a different value of c_i is drawn uniformly at random from the interval $[-\alpha, 1 + \alpha]$, α being a suitable non-negative constant. We used $\alpha = 0.1$.

As a selection operator we chose tournament selection with tournament size 3. To maximally speedup evolution we used a steady state GA. As a replacement strategy we used negative tournament selection.

As a mutation operator we used headless chicken crossover, i.e., the recombination (via BLX) between the individual to be mutated and a randomly generated individual.

Given the complexity of the task we used very large populations including $50,000$ individuals.

While we were able to use a standard representation and standard genetic operators, the design of the fitness function involved much more work and required numerous successive refinements. The fitness function we eventually settled for is described below.

The natural objective function for a mouse is, of course, the extent to which the pointer was moved in the desired direction. So, this is clearly a necessary component in our fitness function. However, this is not enough. For example, it is possible that the pointer moved a great deal in the desired direction, while at the same time drifting significantly in a direction orthogonal to the desired one. A fitness function based on the natural objective would reward this behaviour, effectively leading to very sensitive but not very precise controllers. So, clearly the problem is multi-objective, that is we want to obtain both maximum motion in the desired direction *and* minimal motion in the orthogonal direction.

To deal with this problem we adopted a penalty method to combine the multiple objectives into a single fitness measure. So, the fitness function f is a linear combination of the objectives ω_i of the problem:

$$f = \sum_i \lambda_i \omega_i \tag{2}$$

where λ_i are appropriate coefficients (with λ_1 conventionally being set to 1).

In the mouse control problem we used three different objectives. The first objective, ω_1, assesses the extension of the motion in the target direction, the other two, ω_2 and ω_3, evaluate the motion in orthogonal direction. In particular, ω_2 assesses the average extension of the motion in such a direction at the end of runs. It, therefore, ignores any errors that have been later cancelled by errors

with opposite sign (i.e., in the opposite direction). Instead, ω_3 evaluates the average absolute error deriving from motion orthogonal to the target direction. So, it assesses the extent to which the trajectory towards the target is convoluted.

The performance of the controller was evaluated on the basis of its behaviour over groups of 30 repetitions of a command (up, down, left or right), which we will term *runs*. In order to ensure that the controller performed well in all directions, these runs were acquired for all possible directions (and, in fact, multiple times for each direction to limit risks of over-fitting). All the resulting trials formed the controller's training set.

In each of the examples in the training set the controller produced a velocity vector. Let us call $v^{i,t} = (v_d^{i,t}, v_o^{i,t})$ the velocity vector produced at repetition i in the t-th run. This vector is expressed in a reference system where the component $v_d^{i,t}$ represents the motion in the target direction, while $v_o^{i,t}$ represents the motion produced in the direction orthogonal to the desired direction.

The objectives can then be expressed as as follows:

$$\omega_1 = \nu \sum_{t=1}^{N_r} \sum_{i=1}^{30} v_d^{i,t} \qquad \omega_2 = \nu \sum_{t=1}^{N_r} \sum_{i=1}^{30} v_o^{i,t} \qquad \omega_3 = \nu \sum_{t=1}^{N_r} \sum_{i=1}^{30} |v_o^{i,t}|$$

where N_r is the number of runs and $\nu = \frac{1}{30 \times N_r}$ is a normalisation factor. Since we want to maximise ω_1, but minimise $|\omega_2|$ and ω_3, the penalty coefficients were set as follows: $\lambda_1 = 1$, $\lambda_2 = -0.2$ and $\lambda_3 = -0.2$.

5 Results

5.1 Controlled Experiments

A *4-fold cross-validation* has been applied to train the system and test its performance and generalisation ability. So, for each participant the total of 16 runs has been split in groups of four each containing one run for each direction (1–4, 5–8, etc.). Therefore there were 4 groups for each participant. Then two of these groups have been used as training set while a third one as validation set. For each of the 12 combinations a population (i.e., a different classifier) has been evolved. Runs lasted 40 generations.

The three different objectives (Sect. 4) have been evaluated for the training-set as well as for the validation-set. The analysis of the results (see [3] for details) shows that all participants were able to move the pointer in the target directions (as quantified by ω_1) with very limited lateral error at the end of a sequence of commands (quantified by ω_2). In fact, if we consider ω_1 as the amplitude of the signal and ω_2 as the standard error of the noise in our system, then $\omega_1/|\omega_2|$ gives us an idea of the signal to noise ratio (SNR) during the movement of the pointer. Averaged over our 6 participants, $\omega_1/|\omega_2|$ is 28.1, which gives us a good SNR = 29.0 dB.

These results have been obtained by processing offline data acquired with the BCI mouse in *training* mode. In this mode stimuli are presented to a participant

but no feedback is provided as to the extend and direction of motion of the pointer. This is done to ensure that the cleanest possible dataset is gathered. As soon as the acquisition of an appropriate training set is completed the system performs all the necessary preparatory steps, including filtering and feature detection, for the application of the GA. GA runs are typically successful within 30 generations. The data preparation and the training process require between 5 to 10 minutes. The system can then be used.

5.2 Realistic Applications

Having established that the approach works and appears to be accurate, we decided start exploring more realistic uses of the mouse with participant D – the oldest and worst performing of our subjects. So, the systems we describe below and the results obtained with this participant can only be considered as first, conservative indications of what can be achieved. We will need to perform much deeper investigations of these systems in future research.

The first application we considered was the use of the mouse to control a standard computer system (in particular a personal computer running the Windows XP operating system). In this application, the screen included our four flashing rectangles and a fixation cross as in the tests of the BCI mouse described above. However, naturally, the background was not uniformly gray: instead it included the standard windows and icons normally available in a Windows machine (see Figure 2). In order to make the system as user friendly as possible for people with limited or no saccadic control, instead of moving the mouse pointer on a fixed screen, we decided to scroll the screen, thereby ensuring that the entities of interest for the user were always near the fixation cross. In addition, we used a zoom factor of 2 to ensure maximum readability.

We did not retrain the system with this richer set of stimuli. Nonetheless, the user was immediately able to move the pointer to the desired locations on the screen, albeit slowly at first, despite the large number of moving elements on the screen being a clear distracting factor. After a few minutes the participant had clearly improved his ability to control the system, as shown in the "film strip" in Figure 2, where the participant was able to scroll from the top left corner to the lower left corner of the main window in around 13 seconds, which we find encouraging. We expect that much better performance could be obtained by retraining the control system for this specific application and by giving participants some time to adjust to the system. Also, as we mentioned before, participant D was the worst performing in our study. So, we should expect to see better average performance when looking at a larger set of participants.

Naturally, the availability of 2–D pointer motion makes it possible to input data of any other type. Numerous systems exist, for example, that can turn mouse movements into text. So, as our second test application we designed a simple prototype speller system. We wanted to test the viability of spelling by BCI mouse and determine whether the accuracy of the control is sufficient for the task. In our BCI speller system, in the centre of the screen a circle with 8 sectors is drawn. Each sector represents one of the 8 most used characters in English



Fig. 2. Snapshots (ordered from left to right and from top to bottom) taken every 20 frames (800ms) from a video recording of our BCI mouse in action. The user wanted to scroll down.

(Fig. 3). The idea is for the system to use a T9 cell-phone-type predictor which makes it possible to input complete words with very few key presses. However, we made not use of this feature during our tests.

The system starts with the pointer at the centre of the screen. In order to enter a character, the user moves the cursor in the desired sector of the screen. To give time to the user to move the pointer into the desired sector, character input is prevented for a certain amount of time (15 s in our tests). The character is acquired as soon as the cursor reaches the perimeter of the circle or a maximum time has elapsed. The entered character is appended to the current string shown at the upper right in the screen and the pointer is repositioned at the centre.

The preliminary testing of this system with participant D has been promising, despite, again, our not retraining the system for this application, and the tests being hampered by technical problems. The objective of the tests was to input the sentence “she has a hat”. The user repeated the task three times. The first time the user made 2 errors out of 7 characters inputting characters at a rate of one every 35 seconds. In the second attempt the user was able to spell “she has” without any errors but at a much slower rate (one character every 80 seconds). The participant was able to do a further test spelling again without any errors “she has” plus an additional space, before feeling drained and unable to continue. In this last test the user inputted characters at a rate of one every 54 seconds.

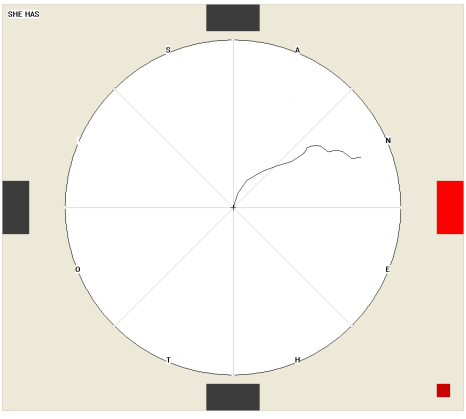


Fig. 3. Our BCI-mouse based speller after the user was been able to correctly enter the text "SHE HAS"

So, despite us running these tests with a user who was tired after the many other tests he had run during his session, after rapidly adjusting to the new application, the user was able to control the device and input data without errors, although this was achieved with a considerable concentration effort and at a very slow rate of 3.2 bits/minute. Again, we should expect much better performance with well-rested and better performing participants. Also, training the control system specifically for this application and giving participants some time to adjust to the system would certainly provide significant performance improvements. We plan to further develop this system in future research.

6 Conclusions

In this paper we have proposed a BCI system based on the use of P300 waves. The system is analogue, in that at no point a binary decision is made as to whether or not a P300 was actually produced in response to a particular stimulus. Instead, the motion of the pointer on the screen is controlled by directly combining the amplitudes of the output produced by a filter in the presence of different stimuli.

Beyond providing carefully designed stimuli, a rich set of features (wavelet coefficients) and a flexible combination mechanism (a filter) through which we thought a solution to the problem of controlling a pointer via EEG could be found, we actually did not do any other design. The biggest part of the design in this system (i.e. the feature selection, the selection of the type, order and parameters of the controller) was entirely left to a genetic algorithm.

The performance of our system has been very encouraging. As mentioned in Sect. 5.1, all participants have been able to use the system within minutes. The GA was very effective and efficient at finding good designs for the system. Indeed, it succeed in every run, suggesting that we had chosen the infrastructure for the system and the feature set reasonably well. In validation, the trajectories

of the pointer have achieved high accuracy. The system issues control commands at a much faster rate (approximately once per second) than other P300-based computer mice previously described in the literature.

These encouraging results indicate that there may be a lot more information about user intentions in EEG signals, and that perhaps, traditional design techniques may be a limiting factor. There is very limited knowledge as to how to manually design analogue BCI mouse systems. Evolution, on the other hand, being entirely guided by objective measures of success (the fitness function), was able to achieve this almost effortlessly. The systems evolved were also rather robust. For example, it was possible to control the mouse even in situations very different from the ones originally considered in training, such as in tests with control in a real Windows environment and in our BCI speller, without retraining the system. So, we suspect that other single-trial BCI applications might benefit from the use of properly guided evolutionary algorithms. We plan to explore this in future research.

References

1. F. Beverina, G. Palmas, S. Silvoni, F. Piccione, and S. Giove. User adaptive BCIs: SSVEP and P300 based interfaces. *PsychNology Journal*, 1(4):331–354, 2003.
2. C. Cinel, R. Poli, and L. Citi. Possible sources of perceptual errors in P300-based speller paradigm. *Biomedizinische Technik*, 49:39–40, 2004. Proceedings of 2nd International BCI workshop and Training Course.
3. L. Citi, R. Poli, C. Cinel, and F. Sepulveda. P300-based brain computer interface mouse with genetically-optimised analogue control. Technical Report CSM-451, Department of Computer Science, University of Essex, May 2006.
4. L. Citi, R. Poli, and F. Sepulveda. An evolutionary approach to feature selection and classification in P300-based BCI. *Biomedizinische Technik*, 49:41–42, 2004. Proceedings of 2nd International BCI workshop and Training Course.
5. E. Donchin and M. G. H. Coles. Is the P300 a manifestation of context updating? *Behavioral and Brain Sciences*, 11:355–372, 1988.
6. J. Donoghue. Connecting cortex to machines: recent advances in brain interfaces. *Nature Neuroscience*, 5:1085–1088, 2002.
7. L. J. Eshelman and J. D. Schaffer. Real-Coded Genetic Algorithms and Interval Schemata. In L. D. Whitley, editor, *Foundations of Genetic Algorithms 2*, San Mateo, CA, 1993. Morgan Kaufmann Publishers.
8. L. A. Farwell and E. Donchin. Talking off the top of your head: A mental prosthesis utilizing event-related brain potentials. *Electroencephalography and Clinical Neurophysiology*, 70:510–523, 1988.
9. J. B. Polikoff, H. T. Bunnell, and W. J. B. Jr. Toward a p300-based computer interface. In *Proc. Rehab. Eng. and Assistive Technology Society of North America (RESNA'95)*, pages 178–180, Arlington, Va, 1995. RESNAPRESS.
10. J. R. Wolpaw, N. Birbaumer, D. J. McFarland, G. Pfurtscheller, and T. M. Vaughan. Brain-computer interfaces for communication and control. *Clinical Neurophysiology*, 113(6):767–791, Jun 2002.
11. J. R. Wolpaw and D. J. McFarland. Control of a two-dimensional movement signal by a noninvasive brain-computer interface in humans. *Proceedings of the National Academy of Sciences*, 101(51):17849–17854, 2004.

A Genetic Programming Approach to Feature Selection and Classification of Instantaneous Cognitive States

Rafael Ramirez and Montserrat Puiggros

Music Technology Group
Universitat Pompeu Fabra
Ocata 1, 08003 Barcelona, Spain
{rafael,mpuiggros}@iua.upf.es

Abstract. The study of human brain functions has dramatically increased in recent years greatly due to the advent of Functional Magnetic Resonance Imaging. This paper presents a genetic programming approach to the problem of classifying the instantaneous cognitive state of a person based on his/her functional Magnetic Resonance Imaging data. The problem provides a very interesting case study of training classifiers with extremely high dimensional, sparse and noisy data. We apply genetic programming for both feature selection and classifier training. We present a successful case study of induced classifiers which accurately discriminate between cognitive states produced by listening to different auditory stimuli.

Keywords: Genetic programming, feature extraction, fMRI data.

1 Introduction

The study of human brain functions has dramatically increased in recent years greatly due to the advent of Functional Magnetic Resonance Imaging (fMRI). While fMRI has been used extensively to test hypothesis regarding the location of activation for different brain functions, the problem of automatically classifying cognitive states has been little explored. The study of this problem is important because it can provide a tool for detecting and tracking cognitive processes (i.e. sequences of cognitive states) in order to diagnose difficulties in performing a complex task.

In this paper we describe a genetic programming approach to detecting the instantaneous cognitive state of a person based on his/her Functional Magnetic Resonance Imaging data. We apply genetic programming for both feature selection and classifier training to the problem of discriminating instantaneous cognitive states produced by different auditory stimuli. We present the results of a case study in which we have trained classifiers in order to discriminate whether a person is (1) listening to a pure tone or a band-passed noise burst, and (2) listening to a low-frequency tone or a high-frequency tone. The problem investigated in this paper is also interesting from the evolutionary computing point

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 311–319, 2007.
© Springer-Verlag Berlin Heidelberg 2007

of view since it provides an interesting case study of training classifiers with extremely high dimensional (10,000-15,000 features), sparse (32-54 examples) and noisy data.

We associate a class with each of the cognitive states of interest and given a subject's fMRI data observed at time t, the classifier predicts one of the classes. We train the classifiers by providing examples consisting of fMRI observations along with the known cognitive state of the subject. We select brain areas by applying a feature selection genetic programming method.

The rest of the paper is organized as follows: Section 2 sets out the background for this research. In Section 3, we describe our approach to the problem of detecting the instantaneous cognitive state of a person based on his/her Functional Magnetic Resonance Imaging data. Section 4 presents a case study and discusses the results, and finally Section 5 presents some conclusions and indicates some areas of future research.

2 Background

2.1 Functional Magnetic Resonance Imaging

Functional Magnetic Resonance Imaging is a brain imaging technique that allows the observation of brain activity in human subjects based on the increase in blood flow to the local vasculature that accompanies neural activity in the brain. It produces time-series data that represents brain activity in a collection of 2D slices of the brain. The collection of the 2D slices form a 3D image of the brain containing in the order of 12000 voxels, i.e. cubes of tissue about 3 millimeters on each side. Images are usually taken every 1-5 seconds. Despite the limitations in temporal resolution, fMRI is arguably the best technique for observing human brain activity that is currently available. Figure 1 shows a fMRI image showing the instantaneous activity of a section of the brain and the activity over time of one of its voxels (white voxels are those with highest activity while dark voxels are those with lowest activity).

Functional Magnetic Resonance Imaging has been widely applied to the task of identifying the regions in the brain which are activated when a human performs a particular cognitive function (e.g. visually recognizing objects). Most of the reported research summarizes average fMRI responses when a human is presented with a particular stimulus repeatedly. Regions in the brain activated by a particular task are identified by comparing fMRI activity during the period where the stimulus is presented with the activity detected under a control condition. Other research describes the effects of varying stimuli on activity, or correlations among activity in different brain regions. In all these cases, the results are statistics of effects averaged over multiple trials and multiple subjects.

Haxby et al [1] detect different patterns of fMRI activity generated when a human views a photograph of different objects (e.g. faces, houses). Although this information was not specifically used for classifying subsequent fMRI data, Haxby et al reported that they could automatically identify the data samples related to the same object category. Wagner et al [2] reported that they have been

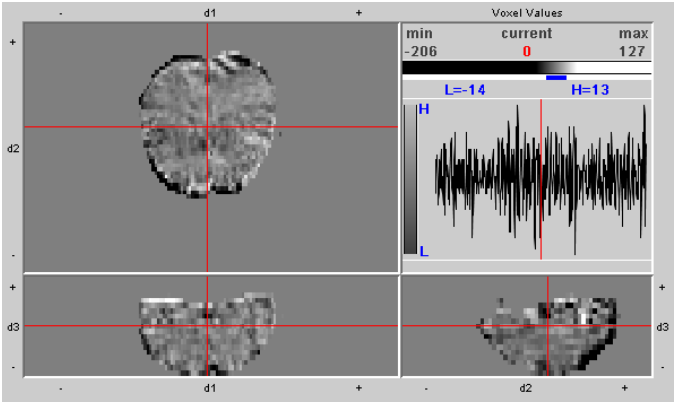


Fig. 1. fMRI image showing the instantaneous activity of a section of the brain and the activity over time of one of its voxels: white voxels are those with highest activity while dark voxels are those with lowest activity

able to predict whether a verbal experience is to be remembered later based on the amount of activity in particular brain regions during the experience. Closer to the work reported in this paper is the work by Mitchell et al [3,4] who have applied machine learning methods to the same problem investigated here. In particular, they have trained classifiers to distinguish whether a subject is looking at a picture or a sentence, reading an ambiguous or non-ambiguous sentence, and the type of word (e.g. a word describing food, people, etc.) to which a subject is exposed. Cox et al [5] applied support vector machine to fMRI data in order to classify patterns of fMRI activation produced by presenting photographs of various categories of objects.

2.2 Genetic Feature Selection

Feature selection [6,7] is a process to select useful features to obtain an efficient and improved solution to a given problem. As the classification task reported in this paper clearly involves a high dimensional training data, it is necessary to apply feature selection methods in order to train the classifiers. Several machine learning techniques have been explored for feature selection. In particular, evolutionary algorithms have also been used for feature selection [8,9,10,11]. Typically, in a genetic algorithm based feature selection approach [8,12], each individual (chromosome) of the population represents a feature subset. For an n-dimensional feature space, each chromosome is encoded by an n-bit binary string $b_1, \ldots b_n$. b_i=1 if the i-th feature is present in the feature subset represented by the chromosome and b_i=0 otherwise. A classifier is used to evaluate each chromosome (or feature subset). Typically each chromosome is evaluated based on the classification accuracy and the dimension of the feature subset (number of 1s). In [13], it has been shown that genetic algorithm based feature selection performs better than many conventional feature selection techniques for high-dimensional data. Siedlecki and

Sklansky [8] used branch and bound technique for feature selection using genetic algorithms. Casillas et al. [9] devised a genetic feature selection scheme for fuzzy rule based classification systems. In [14], ADHOC is a genetic algorithm based feature selection scheme with C4.5 induction learning. Pal et al. [10] proposed a new genetic operator called self-crossover for feature selection. However, there have been only a few attempts to use genetic programming [15,16] for feature selection. In this paper we apply a genetic programming based feature selection algorithm introduced in [17]. We describe the algorithm in the following section.

3 Classifying Cognitive States

In this section we present our approach to training and evaluating classifiers for the task of detecting the instantaneous cognitive state of a person. Given a person's observed instantaneous fMRI data at time t, we train a classifier in order to predict the cognitive state that gave rise to the observed data. The training data is a set of examples of fMRI observations along with the known cognitive state.

3.1 Classifier Training

We are interested in inducing a classifier of the following form:

$$Classifier(fMRIdata(t)) \rightarrow CognState$$

where $fMRIdata(t)$ is an instantaneous fMRI image at time t and $CognState$ is a set of cognitive states to be discriminated. For each subject in the fMRI data sets we trained a separate classifier.

In order to both, perform feature selection and train each separate classifier we apply a multi-tree genetic programming algorithm. For a c-class problem, a population of classifiers, each having c trees is constructed using a randomly chosen feature subset. The size of the feature subset is determined probabilistically by assigning higher probabilities to smaller sizes. The classifiers which are more accurate using a small number of features are given higher probability to evolve, thus encouraging classifiers with a reduced number of features.

For a two-class problem a classifier can be represented by a binary tree T. For a pattern x, if $T(x) \geq 0$, then $x \in class_1$ else $x \in class_2$. For multicategory c-class problems, we usually require a classifier consisting of c such binary trees $T_1, \ldots T_c$. For a pattern x, if $T_i(x) \geq 0$ and $T_j(x) < 0$ for all $j \neq i$, then $x \in class_i$. However, if more than one tree show positive response for a pattern x, then the conflict has to be resolved in order to assign a single class to the pattern. In this paper we are only concerned with two-class classification problems (see case study in section 4), thus we omit the resolution of conflicts in multicategory c-class problems.

Initialization. Each tree of each classifier is randomly generated using the function set $F = \{+, -, *, /\}$, the feature values and a set of randomly generated constants in [0.0, 10.0] (these interval has been selected taking into account the voxels values range in the study reported).

Fitness measure. Each classifiers in the population is trained with the whole set of training samples $X = X_1 \cup \ldots \cup X_c$ where X_k is the set of positive examples for $class_k$. $\mid X_k \mid = N_k$ and $\mid X \mid = N$. The fitness function to compute the fitness of the i-th classifier in the population is as follows:

$$f_i = \sum_{k=1}^{c} \sum_{i=1}^{N_k} \sum_{j=1}^{c} h_{k,i,j}^{l}$$

The two outer summations in are used to consider all data points in X to compute the fitness of a classifier. $h_{k,i,j}^{l}$ is the contribution of j-th tree of l-th classifier T_j^l for the i-th data point x_i of X_k. For a training point $x_i \in X_k$ we expect $T_k^l(x_i) \geq 0$ and $T_{j \neq k}^l(x_i) < 0$. Consequently, both $T_k^l(x_i) \geq 0$ and $T_{j \neq k}^l(x_i) < 0$ should increase the fitness function. To achieve this, $h_{k,i,k}^{l} = (N - N_k)/N_k$ if $T_k^l(x_i) \geq 0$ and $h_{k,i,k}^{l} = 0$ if $T_k^l(x_i) < 0$, and $T_{j \neq k}^l(x_i) = 0$ if $T_{j \neq k}^l(x_i) \geq 0$ and $T_{j \neq k}^l(x_i) = 1$ if $T_{j \neq k}^l(x_i) < 0$.

Selection. We consider the genetic operations: reproduction, crossover and mutation. We use fitness-proportion selection scheme for reproduction operation, tournament selection scheme for crossover operation and random selection scheme for mutation operation.

If some of the trees in a particular classifier are not able to classify properly, then those trees are given more preference to take part in crossover and mutation operations for their improvement. Thus, trees are assigned a mutation and crossover probability p_i in proportion to their fitness. p_i is computed as follows:

$$p_i = k_i / (\sum_{j=1}^{c} k_j)$$

where k_i is the number of training samples not correctly classified by the tree T_i (out of the N training samples).

Crossover and Mutation Operations. As we give more preference to unfit trees to take part in crossover and mutation operations, the chance of evolution of weak trees of potential classifiers is increased and the chance of unwanted disruption of already fit trees is reduced. The detailed steps of the genetic operations are described in [38].

Termination. The genetic programming algorithm is terminated when all N training samples are classified correctly by a classifier in the population or a predefined number M of generations are completed.

3.2 Feature Selection

Selection of a Feature Subset. Let P be the population size. Before initializing each individual classifier C_i, $i \in \{1, \ldots P\}$, a feature subset S_i is randomly chosen from the set of all n available features. The size of the feature subset S_i for each classifier C_i is determined probabilistically. The probability p_r to select a feature subset of size r is defined as

$$p_r = (2/(n-1)) - (2r/n(n-1))$$

p_r decreases linearly with increase in r. Note that $\sum_{r=1}^{n} p_r = 1$. We use Roulette wheel selection to determine the size of the feature subset r with probability p_r.

After deciding the number of features, we randomly select r features to construct the feature subset S_i. The classifier is now initialized as discussed earlier with the chosen feature subset S_i instead of all n features.

Fitness function. The fitness function is required to assign higher fitness value to the classifier which can classify correctly more samples using fewer features. Thus, the fitness function is a multi-objective one, which needs to consider both correct classification and number of features used. It is defined as follows:

$$f_s = f \times (1 + ae^{-r/n})$$

where f is the fitness value obtained by the fitness measure described before, r is the cardinality of the feature subset used, n is the total number of features and a is a parameter which determines the relative importance that we want to assign for correct classification and the size of the feature subset. The factor $e^{-r/n})$ decreases exponentially with increase in r and so is the fitness function. Thus, if two classifiers make correct decision on the same number of training points, then the one using fewer features is assigned a higher fitness. The penalty for using larger feature subset with generations is decreased to improve the classification accuracy. So initially fewer features are used, but as learning progresses more importance to better classification performance is given. To achieve this, a is decreased;

$$a = 2a_f(1 - (gen/M))$$

where a_f is a constant. M is the number of generations, gen is the current generation number. After each generation, the best chromosome is selected using the fitness function:

$$f_s s = f \times (1 + a_f e^{-r/n}).$$

3.3 Classifier Evaluation

We evaluated each induced classifier by performing the standard 2/3 training set 1/3 test set validation procedure in which 33% of the training set is held out for testing while the remaining 67% is used as training data. When performing the validation, we leave out the same number of examples per class. In the data sets, the number of examples is the same for each class considered, thus by leaving out the same number of examples per class we maintain a balanced training set. In order to avoid optimistic estimates of the classifier performance, we explicitly remove from the training set all images occurring within 6 seconds of the hold out test image. This is motivated by the fact that the fMRI response time is blurred out over several seconds.

4 Case Study: Pure Tones and Band-Passed Noise

In this fMRI study [18] twelve subjects (average age = 25; six males) with normal hearing listened passively to one of six different stimulus sets. These sets consisted of either pure tones (PTs) with a frequency of 0.5, 2 or 8 kHz, or band-passed noise (BPN) bursts with the same logarithmically spaced center frequencies and a bandwidth of one octave (i.e. from 0.35-0.7, 1.4-2.8, and 5.6-11.2 kHz, respectively). All stimuli were 500 msec in duration including 50 msec rise/fall times to minimize on/offset artifacts. Stimuli were presented at a rate of 1Hz during the "stimulus-on" intervals of the functional scans. The subjects underwent 12 functional runs consisting of four 32 sec cycles divided into two 16-sec "'stimulus-on" and "stimulus-off" epochs. During six runs each PT and BPN bursts were the on-stimuli. The voxel size was 3.75 x 3.75 x 4.4 mm^3.

We used this data to train classifiers to detect whether a subject is listening to a high or low frequency tone, and whether the subject is listening to a pure tone or a band-passed noise burst. In particular, we trained classifiers for the tasks of distinguishing among the cognitive states for (1) listening to a high frequency PT versus listening to a low frequency PT, (2) listening to a PT versus listening to a BPN burst (both in a middle frequency). Given the general classifier of the form $Cl(fMRIdata(t)) \rightarrow CS$, we are interested in the set CS of cognitive states to be discriminated to be $\{PTHigh, PTLow\}$, and $\{PTMiddle, BPNMiddle\}$.

We initially filter the fMRI data by eliminating voxels outside the brain by discarding the voxels below an activation threshold. The average number of voxels per subject after filtering was approximately 14,000 (it varied significantly from subject to subject). Once the fMRI data is sifted, we further reduced the number of voxels to 2000 by performing a t-test comparing the fMRI activity in the voxel in examples belonging to the two stimuli of interest. We selected 2000 voxels by choosing the ones with larger t-values.

There were a total of 32 training examples available for each subject (i.e. 16 examples for each class). The expected correctly classified instances percentage of a default classifier (selecting the most common class) is 50%. The results obtained were clearly statistically significant and indicate that it is feasible to train successful classifiers to distinguish these cognitive states using the described genetic programming approach. The correctly classified instances percentage for each subject is presented in Table 1.

Table 1. Correctly classified instances percentage for each subject when listening to a high frequency PT versus listening to a low frequency PT (HFPT vs LFPT) and Listening to a middle frequency PT versus listening to a middle frequency BPN burst (MFPT vs MFBP)

Subject	1	2	3	4	5	6
HFPT vs LFPT	86.75	91.67	91.67	87.50	83.63	90.15
MFPT vs MFBP	84.14	88.33	87.50	80.83	82.21	91.67

Discussion. The difference between the results obtained and the accuracy of a baseline classifier, i.e. a classifier guessing at random indicates that the fMRI data contains sufficient information to distinguish these cognitive states, and the described genetic programming method is capable of learning the fMRI patterns that distinguish these states. It is worth noting that the classifier for every subject produced significantly better than random classification accuracies for the study. This supports our statement about the feasibility of training classifiers for the case study reported. However, note that this does not necessary imply that it is feasible to train classifiers for arbitrary tasks. Currently, the general question of exactly which cognitive states can be reliably discriminated remains an open question.

The accuracy of the classifiers for different subjects varies significantly, even within the same study. These uneven accuracies among subjects may be due to the data being corrupted (e.g. by head motion during scanning). In any case, it has been reported that there exists considerable variation in fMRI responses among different subjects.

It is worth mentioning that in all the experiments performed we provided no information about relevant brain regions involved in the tasks performed by the subjects. This contrasts with other approaches (e.g. [3,5]) where the input to the classifiers is the set of voxels in the regions of interests (ROIs) selected for each particular study. Here, we have treated equally *all* voxels in the fMRI studies regardless of which brain region they belong to. Incorporation information about the ROI for each fMRI study would very likely improve the accuracies of the classifiers. We decided not to provide any ROIs information in order to eliminate any feature selection bias and let the genetic programming find the relevant features.

5 Conclusion

In this paper we have explored a genetic programming approach to the problem of classifying the instantaneous cognitive state of a person based on his/her functional Magnetic Resonance Imaging data. The problem provides a very interesting instance of training classifiers with extremely high dimensional, sparse and noisy data. We presented a successful case study of induced classifiers which accurately discriminate between cognitive states involving different types of auditory stimuli. Our results seem to indicate that fMRI data contains sufficient information to distinguish these cognitive states, and that genetic programming techniques are capable of learning the fMRI patterns that distinguish these states. Furthermore, we proved that it is possible to train successful classifiers using features automatically extracted from the studied fMRI data, and with no prior anatomical knowledge. This contrasts previous approaches which consider regions of interest in the brain in order to simplify the problem. As future work, we are particularly interested in interpreting the feature sets used by the classifiers in order to explain the predictions of the classifiers.

Acknowledgments. This work is supported by the Spanish TIC projects ProMusic (TIC2003-07776-C02-01) and ProSeMus (TIN2006-14932-C02-01).

References

1. Haxby, J. et al. (2001) Distributed and Overlapping Representations of Faces and Objects in Ventral Temporal Cortex. Science 2001, 293:2425-2430.
2. Wang, X., et al. (2003) Training fMRI Classifiers to Detect Cognitive States across Multiple Human Subjects. Neural Information Processing Systems.
3. Mitchell, T.M., Hutchinson, R., Niculescu, R.S., Pereira, F., Wang, X., Just, M. and Newman S. (2004). Learning to Decode Cognitive States from Brain Images, Machine Learning, Vol. 57, Issue 1-2, pp. 145-175.
4. Mitchell, T., Hutchinson, R., Just, M., Niculescu, R.N., Pereira, F., Wang, X. (2003). Classifying Instantaneous Cognitive States from fMRI Data, American Medical Informatics Association Symposium, October 2003.
5. Cox, D. D., Savoy, R. L. (2003). Functional magnetic resonance imaging (fMRI) "brain reading": Detecting and classifying distributed patterns of fMRI activity in human visual cortex. NeuroImage, 19, 261-270.
6. M. Dash and H. Liu, (1997). Feature selection for classification, Intell. Data Anal., vol. 1, no. 3, pp. 131-156.
7. A. L. Blum and P. Langley, (1997). Selection of relevant features and examples in machine learning, Artif. Intell. Special Issue on Relevance, vol. 97, pp. 245-271.
8. W. Siedlecki and J. Sklansky, (1989). A note on genetic algorithms for largescale feature selection, Patt. Recognit. Lett., vol. 10, pp. 335-347.
9. J. Casillas, O. Cordon, M. J. Del Jesus, and F. Herrera, Genetic feature selection in a fuzzy rule-based classification system learning process for high-dimensional problems, Inform. Sci., vol. 136, pp. 135-157, 2001.
10. N. R. Pal, S. Nandi, and M. K. Kundu, (1998). Self-crossover: A new genetic operator and its application to feature selection, Int. J. Syst. Sci., vol. 29, no. 2, pp. 207-212.
11. J. Sherrah, R. E. Bogner, and A. Bouzerdoum, (1996). Automatic selection of features for classification using genetic programming, in Proc. Australian New Zealand Conf. Intelligent Information Systems, pp. 284-287.
12. W. Siedlecki and J. Sklansky, (1988). On automatic feature selection, Int. J. Pattern Recognit. Artif. Intell., vol. 2, no. 2, pp. 197-220.
13. M. Kudo and J. Sklansky, Comparison of algorithms that select features for pattern classifiers, Patt. Recognit., vol. 33, pp. 25-41, 2000.
14. M. Richeldi and P. Lanzi, (1996). Performing effective feature selection by investigating the deep structure of the data, in Proc. 2nd Int. Conf. Knowledge Discovery and Data Mining. Menlo Park, CA, pp. 379-383.
15. J. R. Koza, (1992). Genetic Programming: On the Programming of Computers by Means of Natural Selection. Cambridge, MA: MIT Press.
16. W. Banzhaf, P. Nordin, R. E. Keller, and F. D. Francone, Genetic Programming: An Introduction. New York: Morgan Kaufmann.
17. Durga Prasad Muni, Nikhil R. Pal, and Jyotirmoy Das, (2006). Genetic Programming for Simultaneous Feature Selection and Classifier Design IEEE Transactions on Systems, Man and Cybernetics Part B: Cybernetics, vol.36-1.
18. Wessinger, C.M., VanMeter, J., Tian, B., Van Lare, J., Pekar, J., Rauschecker, J.P. (2001). Hierarchical organization of the human auditory cortex revealed by functional magnetic resonance imaging. J Cogn Neurosci. 2001 Jan 1;13(1):1-7.

A Memetic Differential Evolution in Filter Design for Defect Detection in Paper Production

Ville Tirronen, Ferrante Neri, Tommi Karkkainen,
Kirsi Majava, and Tuomo Rossi

Department of Mathematical Information Technology, Agora,
University of Jyväskylä, P.O. Box 35 (Agora), FI-40014 University of Jyväskylä,
Finland
{aleator,neferran,tka,majkir,tro}@jyu.fi

Abstract. This article proposes a Memetic Differential Evolution (MDE) for designing digital filters which aim at detecting defects of the paper produced during an industrial process. The MDE is an adaptive evolutionary algorithm which combines the powerful explorative features of Differential Evolution (DE) with the exploitative features of two local searchers. The local searchers are adaptively activated by means of a novel control parameter which measures fitness diversity within the population. Numerical results show that the DE framework is efficient for the class of problems under study and employment of exploitative local searchers is helpful in supporting the DE explorative mechanism in avoiding stagnation and thus detecting solutions having a high performance.

1 Introduction

In recent years, machine vision systems for quality inspection and fault detection have become popular in paper industry. These systems monitor the paper web for structural defects such as holes and for defects in quality such as faint streaks, thin spots, and wrinkles. Detection of weak defects is crucial in quality paper production since their presence in paper sheets raises difficulties in, for example, printing thus leading to significant monetary losses. The paper web inspection is a challenging task since it must be executed under strict real time constraints. In fact, paper machines can achieve a speed of over 25 m/s, while the defects are barely a few millimeters in size. This poses a serious limitation to applicable techniques and calls for simple filters with high computational efficiency.

In order to pursue this aim, the most commonly used approaches in industrial applications are based on simple threshold techniques, e.g. in [1]. Since these techniques are inadequate for weak defects, more sophisticated methods are required. A popular approach is to utilize texture-based techniques. In [2] the defects are characterized as deviations from the background texture, the problem is encoded as a two class classification problem by means of local binary patterns as source of features and a Self Organizing Map (SOM) as a clustering/classification tool. However, wrinkles and streaks are often quite faint perturbations of texture, and they can be easily missed with texture analysis.

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 320–329, 2007.
© Springer-Verlag Berlin Heidelberg 2007

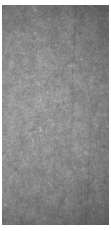


Fig. 1. Source image

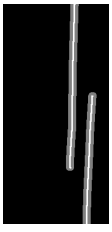


Fig. 2. Label image

This work proposes an approach which employs Finite Impulse Response (FIR) filters, parametrized by a Gabor function, specifically optimized for the task of weak defect detection. The problem oriented design of the FIR filters seems very promising for similar image processing problems with other fields of application, for example in [3] for texture segmentation and in [4] for a vehicle tracking problem. One important difficulty related to this approach is that the problem oriented design of a FIR filter requires the solution of an often challenging optimization problem characterized by a multivariate fitness function. In order to perform the optimization process, in [4] an evolutionary approach has been proposed. The present article proposes a novel implementation of a computational intelligence algorithm in order to design a FIR filter for detecting weak defects in paper production. This paper is organized as follows. In section 2 the filter and problem formulation is presented. Section 3 describes the optimization method which is used in designing the filter according to parameters described in the previous section. Section 4 presents numerical results and section 5 finishes with conclusions of the work.

2 Features of the Filter and Problem Formulation

Inspection of the paper is realized using transillumination. The images acquired in this way contain noisy characteristics as the whole paper structure is imaged. Images may vary by a different background noise field (paper formation) and defect shape. Training of the filter is based on a set of source images supplied by Viconsys Oy[1] and corresponding set of label images. These two sets constitute the training set. Images are taken in a static situation with resolution of approximately 0.6 mm/pixel. Fig. 1 shows an image belonging to the training set with the corresponding label image Fig. 2.

We propose a filter based on the following Gabor function [5]:

$$Gb[\theta, \psi, \sigma_x, \sigma_y, \lambda](x, y) = e^{\frac{(x\cos\theta + y\sin\theta)^2}{2\sigma_x^2}} e^{\frac{(-x\sin\theta + y\cos\theta)^2}{2\sigma_y^2}} \cos\left(\frac{2\pi}{\lambda} + \psi\right), \quad (1)$$

where θ is the angle perpendicular to parallel stripes of the filter, ψ is the phase offset (the filter is symmetrical when $\psi = 0$ and antisymmetrical when $\psi = \frac{\pi}{2}$). Furthermore, σ_x, σ_y specify both the size of the filter and it's ellipticity and λ

[1] `http://www.viconsys.fi/`

is the wavelength of the filter. In other words we use a directed bandpass filter with bandwidth determined by λ and the ratio $\frac{\sigma_x}{\sigma_y}$.

To satisfy real time requirements, we sample the Gabor function into a 7×7 discrete kernel. This will limit the range of possible bandwidth and cause truncation of values, but will retain enough precision and suitable bandwidth for detection of defects of interest.

Since the defects tend to contain curves and the filter is heavily direction dependent, the outputs of two different filters are combined. The coordination of the filters is carried out by assigning a weight coefficient to each filter and then selecting that which produces maximal output for a given pixel. A post-filtering stage, by means of a gaussian filter, is included in order to mitigate the effect of spurious responses due to noise [6]. Thus, for a given image I, α indicating the vector of 12 elements representing the design parameters of the two filters, the filter formula is given by:

$$F(\alpha, I) = G_{15,15} \star max\left[\left(\alpha(1) Gb(\alpha(2), ...\alpha(6)) \star I\right), \left(\alpha(7) Gb(\alpha(8), ...\alpha(12)) \star I\right)\right] \tag{2}$$

where $\alpha(1)$ and $\alpha(7)$ are the weights for both filters, $\star$ denotes the two dimensional discrete convolution and $G_{r,t}$ represents an ordinary gaussian kernel of size $r \times t$. The design parameters of the Gabor filters are shown in Table 1.

Table 1. Design parameters

parameter	description	range of variability
$\alpha(1), \alpha(7)$	weight	$[-100, 100]$
$\alpha(2), \alpha(8)$	θ	$[-\pi, \pi]$
$\alpha(3), \alpha(9)$	ψ	$[0, \pi]$
$\alpha(4), \alpha(10)$	σ_x	$[1, 20]$
$\alpha(5), \alpha(11)$	σ_y	$[1, 20]$
$\alpha(6), \alpha(12)$	λ	$[1, 20]$

The problem of the filter design thus consists of finding a proper set of design parameters α. In order to estimate the fitness of a candidate solution α, the following procedure is carried out. Let us indicate with S the source image and with L the corresponding label image.

The filtered images F are divided into three regions based on the label images: the defect region D defined as the set of those pixels (x, y) of the image F such that $L(x, y) = 1$, the clean region C defined as the set of those pixels (x, y) such that $L(x, y) = 0$, and the non-interesting region characterized by other colors (grey in Fig. 2). Then let us define the similarity function sim as follows:

$$sim(S, L) = a\sigma(D) + b\sigma(C) + c\sqrt{|\mu(D) - \mu(C)|}, \tag{3}$$

where μ and σ denote mean and standard deviation over the set of pixels and a, b, c are weight coefficients. The first term in (3) measures the uniformity of

defect region, the second term measures the noise in background (C region) and the third term measures separation of the two regions. The weight coefficients have been set as $a = 1$, $b = 2$ and $c = -3$ taking into consideration that it is highly desirable the filter clearly separates defects from the background, it is also important that the noise in the background does not lead to detection of false defects. Finally, the fitness function f is given by:

$$f(\alpha) = \frac{1}{n_I} \sum_{k=1}^{n_I} sim\left(F\left(\alpha, S_k\right), L_k\right), \tag{4}$$

where n_I is total number of images in the training set, S_k and L_k are respectively the k^{th} source and label image from the training set. The filter design is thus stated as the minimization of f over $H = [-100, 100]^2 \times [-\pi, \pi]^2 \times [0, \pi]^2 \times [1, 20]^6$.

3 The Memetic Differential Evolution

In order to perform the minimization of f in H a memetic approach is proposed. Memetic algorithms are hybrid algorithms which combine evolutionary algorithms and local searchers [7]. This paper proposes a hybridization of a Differential Evolution (DE) [8] framework with the Hooke-Jeeves Algorithm (HJA) [9] and a Stochastic Local Searcher (SLS) [10]. These local searchers are coordinated by means of a novel adaptive rule which estimates fitness diversity among individuals of the population [11], [12]. DE is a powerful metaheuristic which offers, despite its simple algorithmic philosophy, high performance in many practical cases and has already been widely used in digital filter design (e.g. [13]). Nevertheless, DE is subject to stagnation problems [14]. Thus, our algorithm aims to assist in the explorative power of the DE by hybridizing it with highly exploitative local searchers. The resulting algorithm, namely Memetic Differential Evolution (MDE), consists of the following.

An initial sampling of $S_{pop} = 100$ individuals is executed pseudo-randomly with a uniform distribution function over the decision space H. At each generation, for S_{pop} times, four individuals α_1, α_2, α_3 and α_4 are extracted from the population pseudo-randomly. Recombination according to the logic of a Differential Evolution (DE) occurs at first by generating α'_{off} according to the following formula [15], [8]:

$$\alpha'_{off} = \alpha_1 + K\left(\alpha_2 - \alpha_3\right) \tag{5}$$

where $K = 0.7$ is a constant value set according to the suggestions given in [15]. Then, in order to increase the exploration of this operator, a casuality is introduced by switching some design parameters of α'_{off} with the corresponding genes of α_4. Each switch occurs with a uniform mutation probability $p_m = 0.3$, as suggested in [14], and the offspring α_{off} is thus generated. The fitness value of α_{off} is calculated and, according to a steady-state strategy, if α_{off} outperforms α_4, it replaces α_4, if on the contrary $f(\alpha_{off}) > f(\alpha_4)$, no replacement occurs. The MDE employs the following two local searchers which assist the evolutionary framework (DE) by offering alternative exploratory perspectives.

The **Hooke-Jeeves Algorithm** (HJA) [9], [16] initializes the exploratory radius h_{HJA-0}, an initial candidate solution α and a 12×12 direction exploratory matrix $U = diag\,(w(1), w(2)..w(12))$, where $w\,(m)$ is the width of the range of variability of the m^{th} variable. Let us indicate with $U(m,:)$ the m^{th} row of the direction matrix $m = 1, 2..12$. The HJA consists of an exploratory move and a pattern move. Indicating with α the current best candidate solution and with h_{HJA} the generic radius of the search, the HJA during the exploratory move samples points $\alpha\,(m) + h_{HJA}U(m,:)$ ("+" move) with $m = 1, 2..12$ and the points $\alpha\,(m) - dU(m,:)$ ("-" move) with $m = 1, 2..12$ only along those directions which turned out unsuccessful during the "+" move. If a new current best is found α is then updated and the pattern move is executed. If a new current best is not found, h_{HJA} is halved and the exploration is repeated. The HJA pattern move is an aggressive attempt of the algorithm to exploit promising search directions. Rather than centering the following exploration at the most promising explored candidate solution (α), the HJA tries to move further [17]. The algorithm centers the subsequent exploratory move at $\alpha \pm h_{HJA}U(m,:)$ ("+" or "-" on the basis of the best direction). If this second exploratory move does not outperform $f(\alpha)$ (the exploratory move fails), then an exploratory move with α as the center is performed. The HJA stops when the budget condition of 1000 fitness evaluation is reached.

The **Stochastic Local Searcher** (SLS) [10] picks up a solution α and initializes a parameter σ_{SLS-0}. Then, 24 perturbation vectors h_{SLS} are generated; these vectors having the same length of α and each gene being a random number of a normal distribution having mean value in $\alpha\,(m)$ and standard deviation $\sigma_{SLS}w\,(m)$. Number of perturbations is chosen so that SLS and HJA have similar computational costs. For each of these perturbation vectors, $\alpha + h_{SLS}$ is calculated and the related fitness value is saved. If the most successful perturbation has a better performance than the starting solution, the perturbed solution replaces the starting one, otherwise σ_{SLS} is halved and the process is repeated. The algorithm stops when the budget on the number of fitness evaluations (1000) is exceeded.

In order to perform **coordination of the local searchers**, the MDE uses a novel adaptive scheme. Every 1000 fitness evaluations the following index is calculated:

$$\nu = \min\left\{1, \frac{\sigma_f}{|f_{avg}|}\right\}, \tag{6}$$

where $|f_{avg}|$ and σ_f are respectively the average value and standard deviation over the fitness values of individuals of the population. The parameter ν can vary between 0 and 1 and can be seen as a measurement of the fitness diversity and distribution of the fitness values within the population [18], [11]. More specifically, if $\nu \approx 0$, the fitness values are similar among each other, on the contrary if $\nu \approx 1$, the fitness values are different among each other and some individuals thus perform much better than the others [19], [20]. This index is used to activate local searchers according to the following scheme:

a) if $\nu \geq 0.5$, no local searchers are activated and the DE is continued for 1000 fitness evaluations
b) if $0.4 \leq \nu < 0.5$ one individual of the population is pseudo-randomly chosen and the SLS is applied to it

c) if $0.25 \leq \nu < 0.4$ the SLS is applied to an individual pseudo-randomly chosen and the HJA is applied to that individual having the best performance
d) if $\nu < 0.25$ the HJA is applied to that individual having the best performance

According to our algorithmic philosophy, when ν is small, local searchers are activated in order to offer a different explorative perspective to the DE framework and hopefully detect new promising genotypes. On the other hand, for high values of ν, the DE framework is supposed to exploit the highly diverse population genotypes by recombination. Since the two local searchers have different structures and features, they are activated by means of different threshold values of ν. More specifically, when $0.25 \leq \nu < 0.5$ the DE framework reduces the fitness diversity of the population but still has some genotypes which can likely be exploited. Under this condition, the SLS is supposed to assist the DE attempting to improve available genotypes which can be used by the DE crossover. On the contrary, when ν is very small (e.g. $\nu = 0.2$), fitness diversity is low and thus the end of the optimization process is approaching meaning that the solutions are either all contained in the same basin of attraction or spread out in different areas of the decision space with similar performance [19], [20]. In this case, the SLS is no longer efficient [12]. The HJA is therefore activated in order to execute a deterministic hill-descent of the basin of attraction from the best performing solution with the aim to either include in the population a genotype having high performance or possibly end the game of the optimization process [21].

Finally, ν is also used to initialize the parameters $\sigma_{SLS-0} = \nu/2$ and $h_{HJA-0} = \nu/2$. This choice means that the initial radius of exploration of the local searchers should be large when the fitness diversity is high and small when the fitness diversity is low. More specifically, since the smaller ν is, the nearer the end of the optimization process is (see [19], [20] and [12]), when ν is small the local searchers attempt to detect a better performing solution within the neighborhood of the starting solution, when ν is large the local searchers have a more explorative behavior. The algorithm stops when 200000 fitness evaluations have been performed. The MDE pseudo-code is shown in the following.

```
generate initial population pseudo-randomly;
while budget condition
   initialize fitness counter to 0;
   while fitness counter< 1000
      execute DE recombination and offspring generation;
   end-while
   compute ν = min{ 1, σ_f / |f_avg| };
   if 0.25 ≤ ν < 0.5
      execute SLS on an individual pseudo-randomly selected;
   end-if
   if 0.2 ≤ ν < 0.4
      execute HJA on the individual having the best performance;
   end-if
end-while
```

4 Numerical Results

Performance of the MDE has been compared with the standard Evolution Strategy (ES) according to the implementation proposed in [22], employing the 1/5 success rule shown in [23] and the standard Differential Evolution (DE) [15]. 30 optimization experiments have been performed by means of each of the three algorithms. These three algorithms employ the same population size ($S_{pop} = 100$) and the same budget conditions (200000 fitness evaluations). Table 2 shows results of the optimization process for the three algorithms under study. Table 2 shows the design parameters obtained at the end of the most successful experiments, the corresponding fitness value f^b, the worst (f^w) and the average ($< f >$) fitness values over the 30 experiments carried out and the corresponding standard deviation σ_{exp}.

Table 2. Optimization results

	ES		DE		MDE	
	Filter 1	Filter 2	Filter 1	Filter 2	Filter 1	Filter 2
weight	-5.893	-41.058	2.026	91.448	68.139	-18.267
θ	0.918	2.320	1.014	1.512	1.521	1.763
ψ	1.008	4.622	2.838	4.092	1.724	4.135
σ_x	1.952	5.875	1.000	1.519	7.802	4.713
σ_y	4.552	13.751	1.000	15.737	3.064	3.116
λ	2.330	0.944	0.684	3.522	3.549	1.751
f^b	-1.273		-1.491		-1.526	
$< f >$	-1.139		-1.472		-1.491	
f^w	-1.050		-1.433		-1.462	
σ_{exp}	0.064		0.016		0.013	

Fig. 3 shows the comparison of the algorithmic performance for the three algorithms under study. The average best fitness (see Fig. 3) is the average value over the 30 experiments calculated after every 100 fitness evaluations. Fig. 4 shows the trend of ν vs the number of fitness evaluations in the most successful experiment and highlights the working principles of the MDE.

Numerical results in Table 2 and Fig. 3 show that the DE framework outperforms a classical evolutionary algorithm for the problem under analysis. This result confirms and extends the study in [24] which proves the superiority of the DE with respect to Genetic Algorithms for a similar application. The comparison between the MDE and the DE shows that the MDE is slightly slower in reaching the optimum but eventually outperforms the DE. According to our interpretation, the presence of local searchers softens the explorative feature of the DE framework thus temporarily slowing down the optimization process. On the other hand, local searchers exploit the available genotypes and give a better chance for the DE framework to detect promising search directions. This effect is helpful for avoiding stagnation of the DE and thus converging to a solution having high performance.

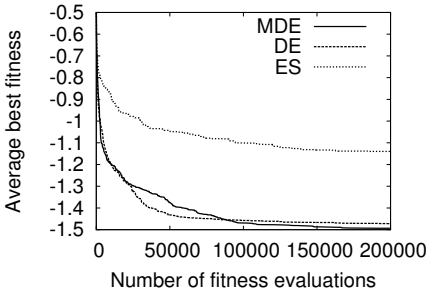


Fig. 3. Algorithmic performance

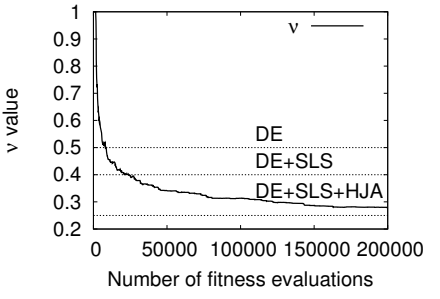
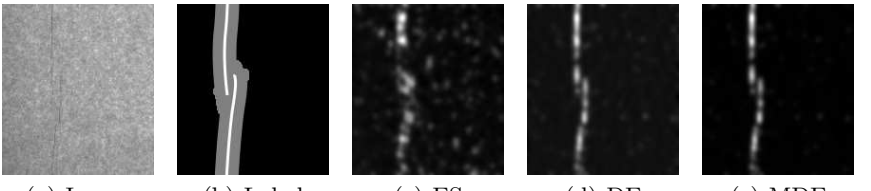
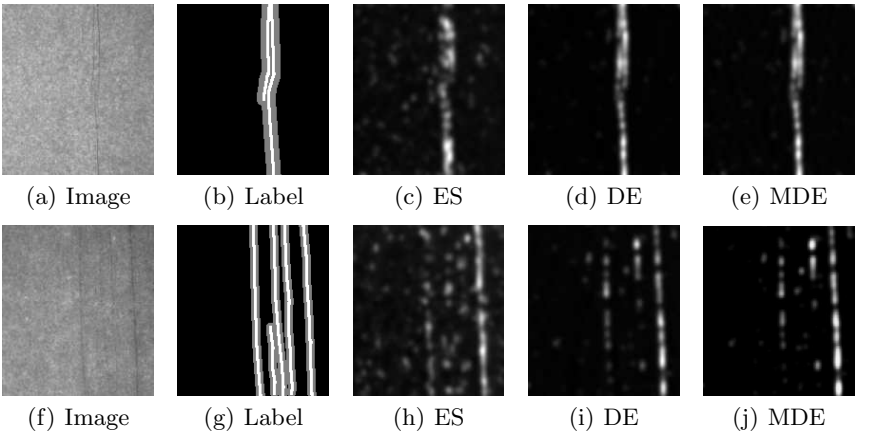


Fig. 4. Trend of ν



(a) Image (b) Label (c) ES (d) DE (e) MDE

Fig. 5. Image belonging to the training set



(a) Image (b) Label (c) ES (d) DE (e) MDE

(f) Image (g) Label (h) ES (i) DE (j) MDE

Fig. 6. Images not belonging to the training set

Figures 5 show an image belonging to the training set, its label image and the filtered images obtained by applying the filters in Table 2. The filters designed by DE and MDE clearly outperform the filter designed by ES. Filters designed by DE and MDE perform, in this case, in a quite similar way but the image filtered by MDE is slightly better in terms of background noise. In order to test a posteriori the general validity of the design, two paper images not belonging to

the training set have been filtered by the filters in Table 2. Results are shown in Fig. 6. The results show that the MDE filter efficiently executes defect detection also working with images not belonging to the training set. As well, the DE and MDE filters clearly outperform that of the ES. In addition, the images filtered by MDE filter are again slightly better than that filtered by the DE filter in terms of background noise and separation of defect and clean regions.

5 Conclusion

This paper proposes a Memetic Differential Evolution (MDE) for designing a digital filter to be employed in detecting weak defects in high quality paper production. MDE is based on an evolutionary framework which uses the exploratory potential of Differential Evolution (DE) and two highly exploitative local searchers which are adaptively activated by means of a novel adaptive rule based on the fitness diversity over the population individuals. Numerical results show that the DE framework performs much better, for this class of problem, than an Evolution Strategy. Moreover the local searchers assist the DE framework in finding solutions having a slightly better performance. The resulting filter is applicable for detecting defects with similar conditions as the training set. In our tests, accuracy of the filter exceeds commonly used methods such as various thresholding schemes and simple edge detectors. Using a miniscule filter kernel leads towards achieving real-time requirements. We believe that this designed filter has industrial applications for detecting streaks and wrinkles.

References

1. Parker, S., Chan, J.: Dirt counting in pulp: An approach using image analysis methods. In: Signal and Image Processing (SIP 2002). (2002)
2. Iivarinen, J., Pakkanen, J., Rauhamaa, J.: A som-based system for web surface inspection. In: Machine Vision Applications in Industrial Inspection XII. Volume 5303., SPIE (2004) 178–187
3. Dunn, D., Higgins, W.: Optimal gabor filters for texture segmentation. IEEE Transactions on Image Processing **4**(7) (1995) 947–964
4. Sun, Z., Bebis, G., Miller, R.: On-road vehicle detection using evolutionary gabor filter optimization. IEEE Transactions on Intelligent Transportation Systems **6**(2) (2005) 125–137
5. Daugman, J.: Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters. **2**(7) (1985) 1160–1169
6. Kumar, A., Pang, G.: Defect detection in textured materials using gabor filters. IEEE Transactions on Industry Applications **38**(2) (2002) 425–440
7. Hart, W.E., Krasnogor, N., Smith, J.E.: Memetic evolutionary algorithms. In Hart, W.E., Krasnogor, N., Smith, J.E., eds.: Recent Advances in Memetic Algorithms, Berlin, Germany, Springer (2004) 3–27
8. Price, K.V., Storn, R., Lampinen, J.: Differential Evolution: A Practical Approach to Global Optimization. Springer (2005)

9. Hooke, R., Jeeves, T.A.: Direct search solution of numerical and statistical problems. Journal of the ACM **8** (1961) 212–229
10. Hoos, H.H., Stützle, T.: Stochastic Local Search Foundations and Applications. Morgan Kaufmann / Elsevier (2004)
11. Caponio, A., Cascella, G.L., Neri, F., Salvatore, N., Sumner, M.: A fast adaptive memetic algorithm for on-line and off-line control design of PMSM drives. IEEE Transactions on System Man and Cybernetics-part B, special issue on Memetic Algorithms 37(1) 2007 28–41.
12. Neri, F., Toivanen, J., Cascella, G.L., Ong, Y.S.: An adaptive multimeme algorithm for designing HIV multidrug therapies. IEEE/ACM Transactions on Computational Biology and Bioinformatics, Special Issue on Computational Intelligence Approaches in Computational Biology and Bioinformatics (2007) to appear.
13. Storn, R.: Designing nonstandard filters with differential evolution. IEEE Signal Processing Magazine **22**(1) (2005) 103–106
14. Lampinen, J., Zelinka, I.: On stagnation of the differential evolution algorithm. In Oŝmera, P., ed.: Proceedings of 6th International Mendel Conference on Soft Computing. (2000) 76–83
15. Storn, R., Price, K.: Differential evolution - a simple and efficient adaptive scheme for global optimization over continuous spaces. Technical Report TR-95-012, ICSI (1995)
16. Kaupe, F., Jr.: Algorithm 178: direct search. Communications of the ACM **6**(6) (1963) 313–314
17. Kelley, C.T. In: Iterative Methods of Optimization. SIAM, Philadelphia, USA (1999) 212–229
18. Neri, F., Cascella, G.L., Salvatore, N., Kononova, A.V., Acciani, G.: Prudent-daring vs tolerant survivor selection schemes in control design of electric drives. In et al., F.R., ed.: Applications of Evolutionary Computing. Volume LNCS 3907., Springer (2006) 805–809
19. Neri, F., Mäkinen, R.A.E.: Hierarchical evolutionary algorithms and noise compensation via adaptation. In Yang, S., Ong, Y.S., Jin, Y., eds.: Evolutionary Computation in Dynamic and Uncertain Environments, Studies in Computational Intelligence, Springer (2007) to appear.
20. Neri, F., Toivanen, J., Mäkinen, R.A.E.: An adaptive evolutionary algorithm with intelligent mutation local searchers for designing multidrug therapies for HIV. Applied Intelligence, Springer (2007) to appear.
21. Eiben, A.E., Smith, J.E.: Hybrid evolutionary algorithms. In: Introduction to Evolutionary Computing, Hybridisation with other Techniques: Memetic Algorithms, Slides of the Lecture Notes, Chapter 10 (2003)
22. Schwefel, H.: Numerical Optimization of Computer Models. Wiley, Chichester, England, UK (1981)
23. Rechemberg, I.: Evolutionstrategie: Optimierung Technisher Systeme nach prinzipien des Biologishen Evolution. Fromman-Hozlboog Verlag (1973)
24. Karaboga, N., Cetinkaya, B.: Performance comparison of genetic and differential evolution algorithms for digital fir filter design. In: Advances in Information Systems LNCS. Volume 3261., Springer (2004) 482–488

Optimal Triangulation in 3D Computer Vision Using a Multi-objective Evolutionary Algorithm

Israel Vite-Silva[1], Nareli Cruz-Cortés[1], Gregorio Toscano-Pulido[2], and Luis G. de la Fraga[1]

[1] CINVESTAV, Department of Computing
Av. IPN 2508. 73060 México, D.F., México
{ivite,nareli}@computacion.cs.cinvestav.mx, fraga@cs.cinvestav.mx
[2] CINVESTAV, Unidad Tamaulipas
Km. 6 carretera Cd. Victoria-Monterrey, 87276, Tamps, México

Abstract. The *triangulation* is a process by which the 3D point position can be calculated from two images where that point is visible. This process requires the intersection of two known lines in the space. However, in the presence of noise this intersection does not occur, then it is necessary to estimate the best approximation. One option towards achieving this goal is the usage of evolutionary algorithms. In general, evolutionary algorithms are very robust optimization techniques, however in some cases, they could have some troubles finding the global optimum getting trapped in a local optimum. To overcome this situation some authors suggested removing the local optima in the search space by means of a single-objective problem to a multi-objective transformation. This process is called *multi-objectivization*. In this paper we successfully apply this *multi-objectivization* to the triangulation problem.

Keywords: Evolutionary Multi-Objective Optimization, 3D Computer Vision, Triangulation.

1 Introduction

One of the foremost 3D Computer Vision problems is how to calculate the three-dimensional reconstruction of a 3D object's visible surface from two images [1,2]. The problem reduces to calculating the three-dimensional point $\mathbf{X}$ that better adjusts to two points $(\mathbf{x}, \mathbf{x}')$ over the images; this problem is known as *triangulation*.

One may project two rays out of two points in known 2D images to intesect inside a reconstructed 3D space. This is not possible in the presence of noise: noise in the location of the points that transmits to the projection matrices (M, M'). Noisless conditions are ideal and rarely to be found in real images making it necessary to look for alternative methods to discover the best point of intersection in 3D space.

The triangulation methods can be applied to three types of reconstructions [2, Ch. 2] namely, the *projective reconstruction* where neither metric nor parallelism

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 330–339, 2007.
© Springer-Verlag Berlin Heidelberg 2007

exists; the *affine reconstruction*, where the concept of parallelism exists but there is no specific metric on each coordinate axis; and the *metric reconstruction*, in which there exists both, a specific metric for each axis and the concept of parallelism. It would be desirable to find a triangulation method which is invariant to any type of reconstruction, this means that under any geometric transformation i.e. translation and rotation, the geometric properties remain unchanged. One option towards achieving this goal is the usage of evolutionary heuristics.

The best known Evolutionary Algorithms are the *Genetic Algorithms* (GA). In general, GAs are very robust optimization techniques capable of finding good solutions even in the presence of noisy search spaces. They are less susceptible to become trapped in local optima than other traditional optimization techniques, due mainly to their stochastic nature and population-based scheme. However, in some cases, if the GA is dealing with a search space composed by a huge number of local optima, it could have some troubles finding the global optimum getting trapped in a local optimum because no small modification of the current GA state will produce a better solution.

In order to overcome this situation, Knowles et, al. [3] suggest to overcome this situation removing the local optima in the search space by means of a single-objective problem to a multi-objective transformation. This process is called *multi-objectivization*. In order to perform the *multi-objectivization* process, it is necessary to replace the original single objective of a given problem with a set of new objectives, or add new objectives in addition to the original function, such that the resulting multi-objective problem has a Pareto optimum front coinciding with the optimum of the original problem. M. T. Jensen [4] showed a successful application of the *multi-objectivization* to solve a very complex problem.

In our case, we applied a single objective Evolutionary Algorithm (EA) to the Triangulation problem obtaining very poor results. As a consequence, we decided to *multi-objectivize* the problem aiming to improve the solutions found.

The main contribution of this paper is show how a representative evolutionary multi-objective state-of-the-art algorithm (NSGA-II [5]) was applied to solve the triangulation problem by *"multi-objectivizing"* the objective function. Our experiments indicate that the results obtained by the Multi-Objective Evolutionary algorithm are much better than those obtained by the single-objective EA. Furthermore, when compared against other known triangulation methods, ours obtains better results if only few correspondence points are available; and a similar performance if the quantity of available points increases.

The rest of the paper is organized as follows: Section 2 shows the triangulation problem; in Section 3 a number of triangulation methods are discussed; Section 4 defines the single and multi-objective optimization problems; in Section 5 the single objective evolutionary triangulation problem is presented; Section 6 shows the multi-objectivized triangulation problem; Section 7 presents the experiments and results; A number of conclusions are drawn in the final Section 8.

2 The Triangulation Problem Statement

Reconstruction is the method by which the spacial layout of a scene and cameras can be recovered from two views [2]. Suppose that the set of 3D points are unknown, and a set of images correspondences $\mathbf{x} \Longleftrightarrow \mathbf{x}'$ are given. The reconstruction goal is to find the camera matrices M and M' and the 3D points $\mathbf{X}_i$ such that:

$$\mathbf{x_i} = M\mathbf{X}_i \text{ and } \mathbf{x}'_i = M'\mathbf{X}_i \text{ for all } i.$$

The reconstruction method follows the next three steps[1]: 1) Compute the fundamental matrix F from point correspondences. This matrix relates both images points and contains the rotation, translation and intrinsec cameras parameters. 2) Compute the projection matrices M and M' from the fundamental matrix. 3) For each point correspondence $\mathbf{x} \Longleftrightarrow \mathbf{x}'$, compute the point in space that projects to these two image points. This third step is known as *triangulation*.

Thus, the *triangulation* method can be thought as the last part of the reconstruction method. If we already have calculated the projection matrices M and M', then we only need to recover the point in three dimensions. However, due to the fact that the points positions over images are inaccuracy plus the noise caused by the inherent propagation of the floating point representation, it is highly possible that the projected lines from the two points in the images do not intersect in the space, then it is necessary to look for a manner to obtain the best approximation. That one is problem addressed in this paper.

3 Methods to Calculate the Triangulation

There exists several triangulation methods [6], however we will describe here only the most relevant ones.

The *middle point triangulation method*, known as *Midpoint*, obtains the middle point over the nearest distance between the two projected lines [7,8]. This method is relatively easy to implement, however its main disadvantage is that it is neither affine nor projective invariant, because in these cases do not exist a defined metric over the angles neither the distances.

The *Linear Least Square method* (LLS) finds an approximation for the triangulation in the least square sense by means of the homogeneous linear equations solution by using the singular value decomposition (SVD). This is affine and Euclidean invariant and its execution time is very low. However, its main weaknesses are: it is not invariant in the presence of a projective reconstruction, and it could be unstable due to the inversion matrix calculation.

The *Poly-Abs* method [6] attempts to minimize the sum of the distances' absolute values between the correspondences points $\mathbf{x} \Longleftrightarrow \mathbf{x}'$ and their corresponding epipolar lines, that is: $d(\mathbf{x}, \lambda) + d(\mathbf{x}', \lambda')$ where λ and λ' are the corresponding epipolar lines. The epipolar lines are computed by using the fundamental matrix. It is affine and projective invariant. This method can find very good results if the fundamental matrix is computed with high precision, if not, the error is very large.

[1] Many variants on this method are possible.

4 Single and Multi-objective Global Optimization

The general global single-objective optimization problem can be defined as follows (let us assume minimization):

Find the vector $\boldsymbol{a}$ such that:

$$\text{minimizes } f(\boldsymbol{a}); \tag{1}$$

where $\boldsymbol{a} = [a_1, ..., a_n]$ is the vector of n decision variables, and the objective function f maps $\boldsymbol{a}$ into $\Re^n \rightarrow \Re$.

The general multi-objective optimization problem, can be defined as follows, Find $\boldsymbol{a}$ such that:

$$\text{minimizes } \boldsymbol{F}(\boldsymbol{a}) \tag{2}$$

where $\boldsymbol{a} = [a_1, ..., a_n]$ is the vector of n decision variables, and $\boldsymbol{F}(\boldsymbol{a})$ is the vector with k objective functions $[f_1(\mathbf{a}), ..., f_k(\mathbf{a})]$.

In general, does not exist a single solution that is minimal for all objectives. Instead, there is a set of solutions P^* called the Pareto optimal set, with the property that:

$$\forall \boldsymbol{a}^* \in P^* \neg \exists \boldsymbol{a} \mid \boldsymbol{a} \preceq \boldsymbol{a}^* \tag{3}$$

where $\boldsymbol{a} \preceq \boldsymbol{a}^* \leftrightarrow (\forall i \in \{1, ..., k\},\ f_i(\boldsymbol{a}) \leq f_i(\boldsymbol{a}^*) \wedge \exists i \in \{1, ..., k\} : f_i(\boldsymbol{a}) < f_i(\boldsymbol{a}^*))$.

The expression $\boldsymbol{a} \preceq \boldsymbol{a}^*$ is read as $\boldsymbol{a}$ dominates $\boldsymbol{a}*$.

In addition, for two solutions $\boldsymbol{a}$ and $\boldsymbol{a}'$, we say $\boldsymbol{a} \sim \boldsymbol{a}'$ if and only if $\exists i \{1, ..., k\}, f_i(\boldsymbol{a}) < f_i(\boldsymbol{a}') \wedge \exists j \in \{1, ..., k\}, j \neq i, f_j(\boldsymbol{a}') < f_j(\boldsymbol{a})$. Such a pair of solutions are said to be incomparable and each is *nondominated* with respect to the other.

Pareto optimal solutions are also termed non-inferior, admissible, or efficient solutions, and their corresponding vectors are nondominated.

5 Applying an Evolutionary Single Objective Algorithm to the Triangulation Problem

In this Section we show how the triangulation problem was adapted to be solved by an evolutionary single-objective optimization algorithm. We experimented with a simple Genetic Algorithm (GA) [9] and a Particle Swarm Optimization Algorithm (PSO) [10] with different parameters values. The statement of the problem and the experiments are presented next.

5.1 Single-Objective Triangulation Problem Definition

The triangulation problem we want to solve can be defined as follows: Find the 3D point $\mathbf{X}$ in world coordinates (X_w, Y_w, Z_w) such that:

Minimizes:

$$f(\mathbf{X}) = d(\mathbf{x}, \hat{\mathbf{x}}) + d(\mathbf{x}', \hat{\mathbf{x}}'), \tag{4}$$

where:

d(*,*) are Euclidean distances, $\mathbf{x}$ and $\mathbf{x}'$ represent the correspondence 2D points from the first and second images respectively, $\hat{\mathbf{x}}$ and $\hat{\mathbf{x}}'$ are 2D reconstructed (estimated) points from the first and second images respectively. $\hat{\mathbf{x}}$ has coordinates $(\hat{x}, \hat{y})$ calculated by:

$$\hat{x} = \hat{u}/\hat{w}, \qquad \hat{y} = \hat{v}/\hat{w}, \qquad [\hat{u}, \hat{v}, \hat{w}]^T = M\,[X_w, Y_w, Z_w, 1]^T \tag{5}$$

$\hat{\mathbf{x}}'$ has coordinates $(\hat{x}', \hat{y}')$ computed by

$$\hat{x}' = \hat{u}'/\hat{w}', \qquad \hat{y}' = \hat{v}'/\hat{w}', \qquad [\hat{u}', \hat{v}', \hat{w}']^T = M'\,[X'_w, Y'_w, Z'_w, 1]^T \tag{6}$$

M and M' are the 3×4 projection matrices from the first and second images respectively.

5.2 Experiments for the Single-Objective Triangulation Problem

The first set of experiments was to execute a Genetic Algorithm (GA) [9] and a Particle Swarm Optimization algorithm (PSO) [10] attempting to minimize Equation (4).

For all the performed experiments we used synthetic images generated by projecting a regular polyhedron. In order to set the correspondence points $\mathbf{x} \Longleftrightarrow \mathbf{x}'$ we selected 24 pair points from each image. Then each point was perturbed with bi-dimensional Gaussian noise (k RMS pixels in each axial direction) with zero mean and standard deviation of k pixels for k from 1 to 8.

Actually, we experimented with a large set of different parameters' values and operators, however, due to space reasons, only some of them are presented here.

The average results from 30 independent runs for both GA and PSO algorithms, are presented in Figure 1. They are compared against the Linear Least Square (LLS) and PolyAbs methods [6] which are one of the best known methods. Only eight pair points out of the 24 available ones were randomly selected to compute the fundamental matrix.

The parameters' values used by the GA were the following: Representation= Binary with Gray Codes, Selection Type=Binary Tournament, Crossover Type= Two Points, Number of Generations=20000, Population Size=100.

The parameters' values applied to the PSO were the following: Number of Generations= 1000000, Population Size= 50, C1=1.4962, C2=1.4962, W=0.7298

It is easy to see that these results are not competitive at all. This failure was our main reason for applying multi-objectivization to the problem. The corresponding experiment is presented in next section.

6 Applying a Multi-objective Evolutionary Algorithm to the Multi-objectivized Triangulation Problem

Due to the disappointing results obtained by the single objective evolutionary algorithms (GA and PSO) we decided to *multi-objetivize* the problem [3,4].

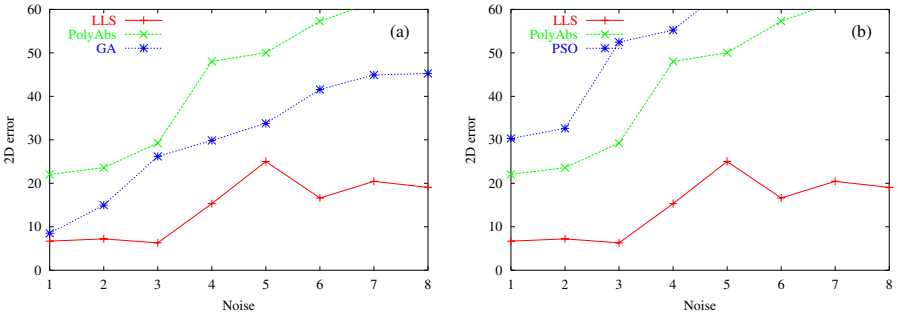


Fig. 1. Comparing the LLS and Poly-Abs methods under a Projective reconstruction against (a) Genetic Algorithm and (b) the Particle Swarm Optimization

For the multi-objectivization sake, the triangulation problem can be formulated as a two-objective optimization problem decomposing the original function in the following way:

6.1 Two-Objective Triangulation Problem Definition

Find the 3D point $\mathbf{X}$ in world coordinates (X_w, Y_w, Z_w) such that:

Minimizes:

$$f_1(\mathbf{X}) = d(\mathbf{x}, \hat{\mathbf{x}}) \qquad \text{and} \qquad f_2(\mathbf{X}) = d(\mathbf{x}', \hat{\mathbf{x}}') \tag{7}$$

where: d(*,*) are Euclidean distances, $\mathbf{x}$ and $\mathbf{x}'$ represent the correspondence 2D points from the first and second images respectively, $\hat{\mathbf{x}}$ and $\hat{\mathbf{x}}'$ are 2D reconstructed (estimated) points from the first and second images respectively. $\hat{\mathbf{x}}$ has coordinates $(\hat{x}, \hat{y})$ calculated by Equation (5) and $\hat{\mathbf{x}}$ has coordinates $(\hat{x}, \hat{y})$ computed by Equation (6).

6.2 The NSGA-II Algorithm

The Evolutionary Multi-Objective Optimization area (EMOO) is a very active one[2], numerous algorithms are published every year [11,12,13].

The NSGA-II [5] is a representative state-of-the-art multiobjective optimization algorithm and one of the most competitive to date. We applied the NSGA-II to solve our *multi-objectivized* problem[3] shown in Eq. (7).

Our two-objective optimization problem was fitted to the NSGA-II in the following manner:

The three coded variables for each individual are $\mathbf{X} = (X_w, Y_w, Z_w)$ by using binary representation. The individuals' fitness is equal to the objective functions

[2] An updated EMOO repository is available at: http://delta.cs.cinvestav.mx/~ccoello/EMOO/

[3] The last version of this software is available at: http://www.iitk.ac.in/kangal/codes.shtml

$f_1(\mathbf{X})$ and $f_2(\mathbf{X})$ (Eq. 7). The parameters' values are the following: Number of generations=300, Crossover rate=0.9, Mutation rate=0.33, Population size= 100, Chromosome's length=106.

The multi-objective problem has Pareto front coinciding with the single-objective optimum, then the solution to our original single-objective problem is taken precisely from the Pareto set. We measure the single-objective function (Eq. 4) in all the Pareto front points. Then, that solution with the less error is selected.

7 Experiments of the Multi-objectivized Triangulation Problem

For all experiments we used synthetic images generated projecting a regular polyhedron. In order to set the correspondence points $\mathbf{x} \Longleftrightarrow \mathbf{x}'$, we selected 24 point pairs from each image. Then each point was perturbed with bi-dimensional Gaussian noise (k RMS pixels in each axial direction) with zero mean and standard deviation of k pixels with k from 1 to 8. The Projective and Affine reconstructions were tested. For each of them, we experimented by taken randomly different quantities of pair of points to compute the fundamental and projection matrices, namely 8, 12, 16 and 20 point pairs out of the 24 available ones.

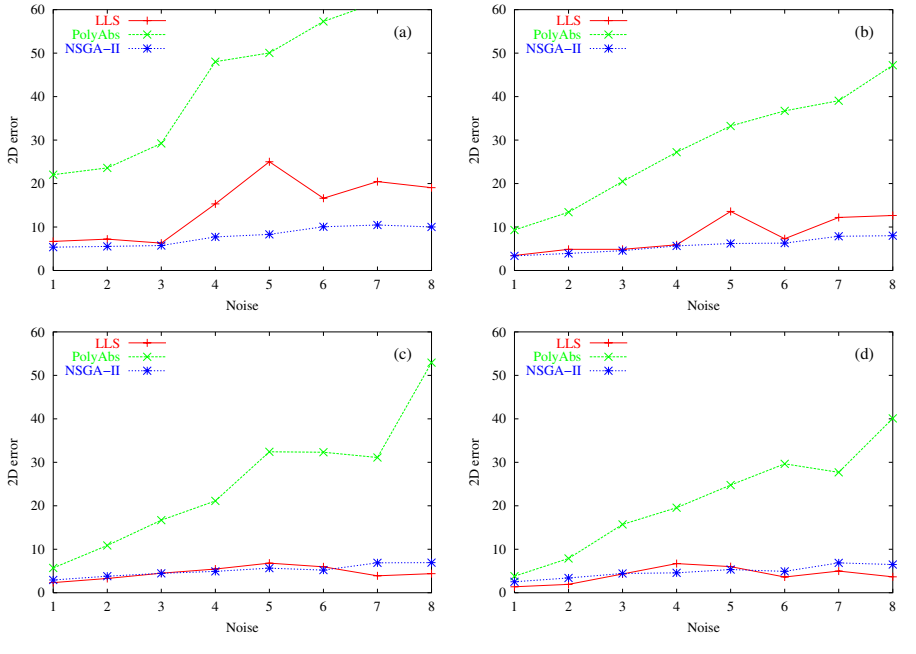


Fig. 2. Comparing the NSGA-II, LLS and Poly-Abs methods under a Projective reconstruction. The number of point pairs taken to calculate the fundamental matrix are: (a) eight pairs, (b) twelve, (c) sixteen and (d) twenty.

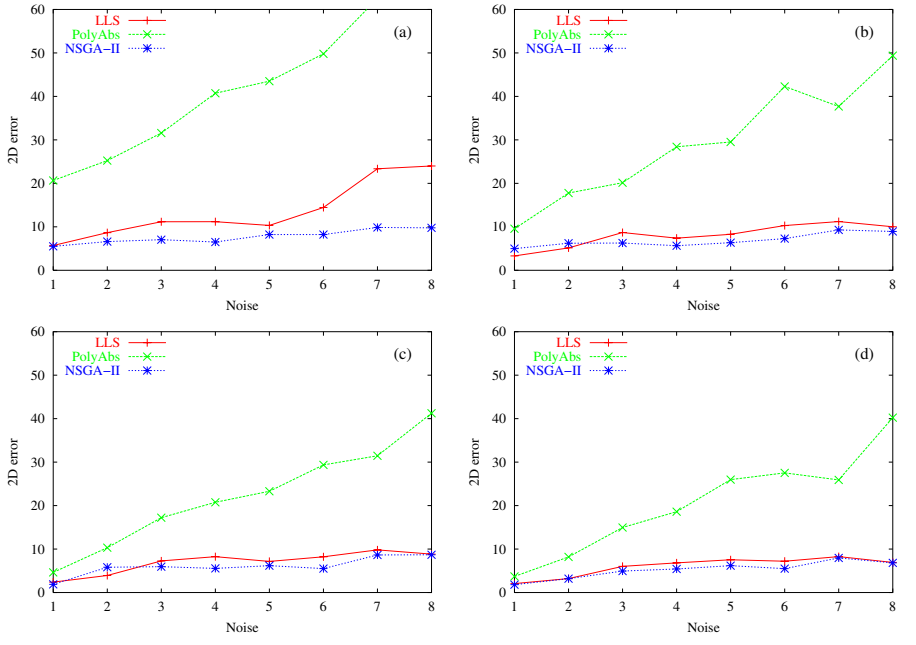


Fig. 3. Comparing the NSGA-II, LLS and Poly-Abs methods under a Affine reconstruction. The number of point pairs taken to calculate the fundamental matrix are: (a) eight pairs, (b) twelve, (c) sixteen and (d) twenty.

The average results from 30 independent executions of the algorithm with noisy points from 1 to 8 pixels, taking 8, 12, 16 and 20 correspondence points are shown in Figures 2 and 3 under Projective and Affine reconstructions, respectively. These results are compared against the LLS and Poly-Abs methods [6].

For the experiment under projective reconstruction, if we only take 8 and 12 correspondence points to compute the fundamental matrix (Fig. 2(a) and (b)) it can be observed that the NSGA-II clearly outperforms LLS and Poli-Abs methods. The 2D error decreases when the noise increases. However, when taking 16 and 20 correspondence points (Fig. 2(c) and (d)) the NSGA-II's advantage is lost with respect to the LLS method showing very similar performances. With respect to the experiment under an affine reconstruction with 8 correspondence points (Fig. 3(a)), NSGA-II is clearly better than the other two methods. For the 12 and 16 correspondence points cases (Fig. 3(b) and (c)) NSGA-II is still better, but the advantage is reduced with respect to the LLS method. And when taking 20 correspondence points, it seems to be that NSGA-II and LLS show similar results.

For all cases Poly-Abs presents a poor performance.

7.1 Discussion

The results obtained by the multi-objectivized problem show that the Evolutionary Algorithm outperforms the LLS and Poly-Abs triangulation methods if

there are only a few quantity of correspondence points. However, if more correspondence points are available, then the LLS and the EA performances are very similar. These correspondence points are difficult to compute specially for real images.

On the other hand, it is important to note that for all the experiments presented in this paper, we assume that the correspondence points $\mathbf{x} \Longleftrightarrow \mathbf{x}'$ from the two images are noisy, which is actually a realistic situation in almost all the real images cases. This, of course, implies that the fundamental and projection matrices are noisy too.

Furthermore, both the single and the multi-objectivized triangulation problems using an EA are invariant to all the types of reconstruction.

The multi-objectivized approach can be successful applied if there are only few very noisy correspondence points and the algorithm execution time is not important, only the obtained accuracy.

8 Conclusions

We have shown a number of experimental results obtained by a single-objective Evolutionary Algorithms to the Triangulation problem, they present a very large error. These disappointing results were dramatically improved when the *multi-objectivization* was applied to the problem and solved by the NSGA-II algorithm.

The *multi-objectivized* approach can be successful applied if there are only few very noisy correspondence points obtaining better results than other known triangulation techniques.

Based on our experiments, we can argument that the *multi-objectivization* methodology performs quite well for the presented triangulation problem. It is worth to remark that such application problems are rather rare.

To the best of our knowledge, there are only other two approaches in the specialized literature reporting the successful application of the multi-objectivization methodology, they are [3] (where it was originally proposed) and [4].

Acknowledgments. This work was partially supported by CONACyT, México, under grant 45306 and scholarship 173964.

References

1. T. Jebara, A. Azarbayejani, and A. Pentland. 3D Structure from 2D Motion. *IEEE Signal Processing Magazine*, 6(3):66–84, May 1999.
2. R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision.* Cambridge University Press, ISBN: 0521540518, 2nd edition, 2004.
3. J. D. Knowles, R. A. Watson, and D. W. Corne. Reducing Local Optima in Single-Objective Problems by Multi-Objectivization. In E. Zitzler, K. Deb, L. Thiele, C. A. Coello Coello, and D. Corne, editors, *Proceedings of the First International Conference on Evolutionary Multi-Criterion Optimization (EMO 2001)*, volume 1993 of *LNCS*, pages 269–283, Berlin, 2001. Springer-Verlag.

4. M. T. Jensen. Guiding Single-Objective Optimization Using Multi-objective Methods. In G. Raidl et al., editor, *Applications of Evolutionary Computing. Evoworkshops 2003: EvoBIO, EvoCOP, EvoIASP, EvoMUSART, EvoROB, and EvoSTIM*, pages 199–210, Essex, UK, April 2003. Springer. Lecture Notes in Computer Science Vol. 2611.
5. K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, April 2002.
6. R. Hartley and P. Sturm. Triangulation. *Computer Vision and Image Understanding*, 68(2):146–157, 1997.
7. P. A. Beardsley, A. Zisserman, and D. W. Murray. Navigation Using Affine Structure and Motion. In J. O. Eklundh, editor, *Proc. 3rd European Conference on Computer Vision - ECCV'94*, volume 800 of *LNCS*, pages 85–96. Springer-Verlag, 1994.
8. P. A. Beardsley, A. Zisserman, and D. W. Murray. Sequential Updating of Projective and Affine Structure from Motion. *International Journal of Computer Vision*, 23(3):235–259, 1997.
9. A. E. Eiben and J. E. Smith. *Introduction to Evolutionary Computing*. Springer, Berlin, 2003.
10. J. Kennedy and R. C. Eberhart. *Swarm Intelligence*. San Mateo, CA: Morgan Kaufmann Publishers, 2001.
11. D. W. Corne, N. R. Jerram, J. D. Knowles, and M. J. Oates. PESA-II: Region-Based Selection in Evolutionary Multiobjective Optimization. In L. Spector, E. D. Goodman, A. Wu, W. B. Langdon, H. M. Voigt, M. Gen, S. Sen, M. Dorigo, S. Pezeshk, M. H. Garzon, and E. Burke, editors, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)*, pages 283–290, San Francisco, California, USA, 7-11 2001. Morgan Kaufmann.
12. K. Deb, M. Mohan, and S. Mishra. Evaluating the Epsilon-Domination Based Multi-Objective Evolutionary Algorithm for a Quick Computation of Pareto-Optimal Solutions. *Evolutionary Computation*, 13(4):501–525, 2005.
13. L. V. Santana-Quintero and C. A. Coello Coello. An Algorithm Based on Differential Evolution for Multi-Objective Problems. *International Journal of Computational Intelligence Research*, 1(2):151–169, 2005.

Genetic Programming for Image Recognition: An LGP Approach

Mengjie Zhang and Christopher Graeme Fogelberg

School of Mathematics, Statistics and Computer Sciences
Victoria University of Wellington, P. O. Box 600, Wellington, New Zealand
{mengjie,fogelbch}@mcs.vuw.ac.nz

Abstract. This paper describes a linear genetic programming approach to multi-class image recognition problems. A new fitness function is introduced to approximate the true feature space. The results show that this approach outperforms the basic tree based genetic programming approach on all the tasks investigated here and that the programs evolved by this approach are easier to interpret. The investigation on the extra registers and program length results in heuristic guidelines for initially setting system parameters.

1 Introduction

Image recognition tasks occur in a wide variety of problem domains. While human experts can frequently accurately classify and recognise objects and images manually, such experts are typically rare or too expensive. Thus computer based solutions to many of these problems are very desirable.

Since the 1990s, Genetic Programming (GP) [1,2] has been applied to a range of image recognition tasks [3,4,5,6] with some success. While showing promise, current GP techniques frequently do not give satisfactory results on difficult image recognition tasks, particularly those with multiple classes (tasks with more than two classes). There are at least two limitations in current GP program structures and fitness function used in these classification systems that prevent GP from finding acceptable programs in a reasonable time.

The programs that GP evolves are typically tree-like structures [7], which map a vector of input values to a single real-valued output[8,9,10,11]. For image recognition/classification tasks, this output must be translated into a set of class labels. For binary classification problems, there is a natural mapping of negative values to one class and positive values to the other class. For multi-class classification problems, finding the appropriate boundaries on the numeric value to well separate the different classes is very difficult. Several new translation rules have recently been developed in the interpretation of the single output value of the tree based GP [9,12,13], with differing strengths in addressing different types of problem. While these translations have achieved better classification performance, the evolution is generally slow and the evolved programs are still hard to interpret, particularly for more difficult problems or problems with a large number of classes.

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 340–350, 2007.
© Springer-Verlag Berlin Heidelberg 2007

In solving image recognition problems, GP typically uses recognition accuracy, error rate or a similar measure as the fitness function [10,12,13], which approximates the true fitness of an individual program. Given that the training set size is often limited, such an approximation frequently fails to accurately estimate the classification of true feature space.

To avoid these problems, this paper aims to investigate an approach to the use of linear genetic programming (LGP) structure and a different fitness function for multi-class image recognition problems. This approach will be examined and compared with the basic tree based GP (TGP) approach on three image classification tasks of increasing difficulty. We will investigate whether the LGP approach outperforms the basic TGP approach in terms of both recognition accuracy and the comprehensibility of the evolved genetic programs. We will also check the effect of the program length and extra registers in the LGP approach.

2 LGP for Multi-class Image Recognition

We used the idea of the register machine LGP [2] and developed a new LGP package recently [14]. In this LGP system, an individual program is represented by a sequence of register machine instructions, typically expressed in human-readable form as C-style code. Prior to any program being executed, the registers which it can read from or write to is zeroed. The features representing the objects to be classified are then loaded into predefined positions in the registers. The program is executed in an imperative manner and can represent a *graph*, which is different from TGP that represents a tree. Any register's value may be used in multiple instructions during the execution of the program.

2.1 Multiple Registers for Multiple Classes

An LGP program often has only one register interpreted in determining its output [15,2]. This kind of form can be easily used for regression and binary classification problems as in TGP.

In this work, we use LGP for multi-class image recognition problems, where we want an LGP program to produce multiple outputs. Instead of using only one register as the output, we use multiple registers in a genetic program, each corresponding to a single class. For a program with an object image as input, the class represented by the register with the largest value is considered the class of the input object image. This is very similar to a feed forward neural network classifier [16]. However, the structure of such an LGP program is much more flexible than the architecture of the neural networks.

2.2 Genetic Operators

We used reproduction, crossover and mutation as genetic operators. In reproduction, the best programs in the current generation are copied into the next generation without any change. Two different forms of mutation [17] were used

in this work. *Macromutation* replaces of an entire instruction with a randomly generated one. *Micromutation* changes just one part of an instruction — either the destination register, a source register or an operation. In the crossover operator, a part from each of the two parent programs is selected and the two parts are swapped to produce offspring. While this is similar to the two-point crossover in genetic algorithms[18], the two selected parts can have different lengths here.

2.3 Fitness Function

Since the size of the training set is finite, any fitness function can only be an approximation to program's true fitness. In an image recognition problem, a program's true fitness is the fraction of the feature space it can correctly classify. A good fitness function is one which accurately estimates this fraction.

A commonly used fitness function for image recognition problem is the *error rate* (or *recognition rate*) of a program recogniser. While it performs reasonably well on some problems, this fitness function frequently fails to accurately estimate the fraction of the feature space correctly classified by a program.

Figure 1(a) shows a simple classification problem with two features `f1` and `f2`. Figure 1(a1) shows the true feature space — feature vectors representing class c1 objects always appear in the fraction of the feature space denoted "c1", and similarly for the fractions denoted "c2" and "c3". Figure 1(a2) shows that `program1` misclassifies two objects of c2 as c3. This program has an error rate of 15% (2/13). Figure 1(a3) shows that `program2` misclassifies one object from class c3 and one object from class c1 as class c2. This program also has an error rate of 15% and will be treated the same as the `program1`. As shown in the two diagrams, `program2` actually classified a larger fraction of the true feature space and approximated the true fitness more accurately than `program1`, but the fitness function cannot accurately reflect this difference.

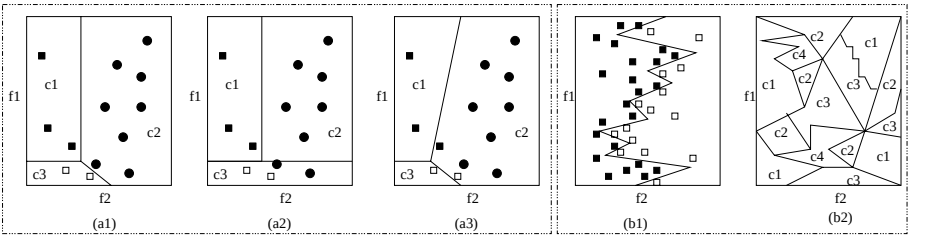


Fig. 1. Program fitness vs true feature space

We call this problem *the hurdle problem*, and it usually occurs when (any two) classes have a complex boundary in the feature space, such as those shown in figure 1 (b1) and (b2). In such a situation, it is easy to classify the bulk of fitness cases for one class correctly, but learning to recognise the other class often initially comes only at an equal or greater loss of accuracy in classifying the first

class. This is a kind of local optimum and GP with such a fitness function often cannot surmount the hurdle within a limited number of generations.

To avoid the hurdle problem, we introduced a new fitness function, f, to more accurately measure how well an individual program classifies the feature space. The new fitness function uses an *increasing penalty* for each of the M_c misclassifications of some class c, as shown in equation 1.

$$f = \sum_{c} \sum_{i=0}^{M_c} \alpha^{\beta i} \tag{1}$$

where α and β are constants with $\alpha > 1$ and $\beta > 0$ to guarantee that the penalty of later misclassifications can be exponentially increased. As α approaches 1.0, f becomes more and more similar to the commonly used fitness function (error rate in this case) in image recognition/classification.

3 Image Data Sets and Experiment Configuration

3.1 Data Sets

Experiments were conducted on three different image data sets providing image classification problems of increasing difficulty. Sample images for each data set are show in figure 2.



Fig. 2. Image data sets. (a) shape; (b) digit01; (c) digit02.

The first data set (*shape*, figure 2a) was generated to give well defined objects against a relatively clean background. The pixels of the objects were produced using a Gaussian generator with different means and variances for each class. Four classes of 600 small objects (150 for each class) formed the data set. The four classes in the left to right order are: dark circles (C1), light circles (C2), dark squares (C3), and light squares (C4). Note that the objects between classes C1 and C3 and between classes C2 and C4 are very similar in the average value of total pixels, which makes the problem reasonably difficult.

The second and third data sets contain two digit recognition tasks, each consisting of 1000 digit examples. Each digit is represented by a 7×7 bitmap image. In the two tasks, the goal is to automatically recognise which of the 10 classes

(digits 0, 1, 2, ..., 9) each digit example belongs to. Note that all the digit patterns have been corrupted by noise. In the two tasks (figure 2 (b) and (c)), 15% and 30% of pixels, chosen at random, have been flipped. In data set 2 (*digit1*), while some patterns can be clearly recognised by human eyes such as "0", "1", "4", "2" and possibly "7", it is not easy to distinguish between "3", "8" and "9", even "5" and "6". The task in data set 3 (*digit2*) is even more difficult — human eyes cannot recognise majority of the patterns. In addition, the number of classes is much larger than that in task 1 and the number of features is very large, making the two tasks even more difficult.

3.2 Terminal Set and Function Set

In the *shape* data set, we used eight statistical features (`f1, f2, ..., f8` corresponding to the source registers `cf[0], cf[1], ..., cf[7]` in LGP) extracted from different parts of the object images and an random number as the terminal set. For the two digit data sets, we used the raw pixels as the terminal sets, meaning that 49 feature terminals were used. The large number of terminals makes the two digit recognition problems more difficult, and we expect the LGP system to automatically select those highly relevant to each recognition problem.

The function set for all the three data sets was {`+, -, *, /, if`}. Division (/) was protected to return 0 on a divide-by-zero. `if` executes the next statement if the first argument is less than the second, and does nothing otherwise.

3.3 Parameters and Termination Criteria

The parameter values used for the LGP system for the three data sets are shown in table 1. The evolutionary process is terminated at generation 50 unless a successful solution is found, in which case the evolution is terminated early.

Table 1. Parameter values for the LGP system for the three data sets

Parameter	shape	digit01	digit02	Parameter	shape	digit01	digit02
pop_size	500	500	500	macromutation_rate	30%	30%	30%
max_program_length	16	40	40	micromutation_rate	30%	30%	30%
reproduction_rate	10%	10%	10%	α	1.15	1.15	1.15
crossover_rate	30%	30%	30%	β	0.18	0.18	0.18

3.4 TGP Configuration

The LGP approach developed in this work will be compared to the TGP approach [7]. In TGP, the ramped half-and-half method was used for program generation [2]. The proportional selection mechanism and the reproduction, crossover and mutation operators [7] were used in evolution. The program output was translated into a class label according to the static range selection method [9]. The TGP system used the same terminal sets, function sets, fitness function,

population size and termination criteria for the three data sets as the LGP approach. The reproduction, mutation, and crossover rates used were 10%, 30%, and 60%, respectively. The program depth was 3–5 for the shape data set, and 4–6 for the two digit data sets.

Notice that the program depths above in TGP were derived from the LGP program lengths based on the following heuristic: an LGP instruction typically consists of one or two arguments and an operation, each corresponding to a node in a TGP program tree. Thus an LGP instruction might initially seem to be equivalent to 2–3 nodes. Considering that each TGP operation might be used by its children and parents, an LGP instruction is roughly corresponding to 1.5 tree nodes. Assuming each non-leaf node has two children, we can calculate the capacity of a depth-n TGP from LGP program instructions.

4 Results and Discussion

4.1 Image Recognition/Classification Performance

All single experiments were repeated 50 runs and the average results are presented here. Figure 3 shows a comparison between the LGP approach developed in this work and the standard TGP approach for the three image recognition problems.

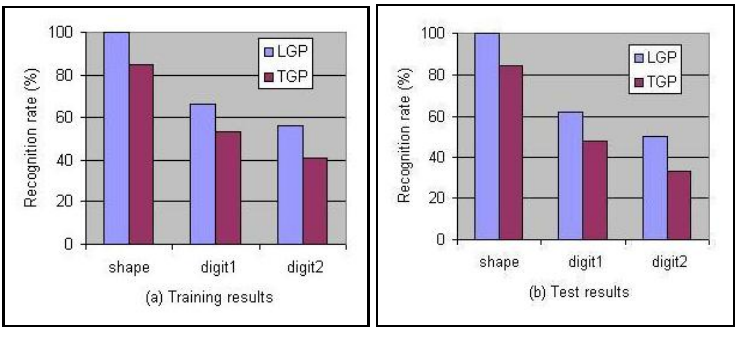


Fig. 3. Recognition rates of the LGP and TGP on the three image data sets

As can be seen from the figure, the LGP approach presented in this paper always achieved better recognition accuracy than TGP on these data sets. On the relatively easy problem in the shape data set, the LGP approach achieved almost ideal performance on both the training and the test sets, which is much better than the TGP approach (about 84%). On the difficult digit data sets with a large number of classes and a very high number of dimensions of features, the LGP approach achieved even better results than humans. On all the data sets, the improvements of LGP over TGP were over 10%, which is quite considerable. In fact, we also did experiments on three subsets of the Yale Faces Database B

[19], and the results (not presented here due to page limitation) showed a similar pattern. This suggests that the LGP approach outperforms the TGP approach for these multi-class image recognition/classification problems.

4.2 Understandability of the Evolved Genetic Programs

To check understandability of the evolved genetic programs by the LGP approach, we used a typical evolved program which perfectly classified all object images for the shape data set as an example. The core code of the evolved genetic program with structural introns commented using `//` is shown in figure 4 (left). The graph representation of the program is shown in figure 4 (right) after the introns are removed. In the figure, the filled circles are output class labels corresponding to the destination registers in the LGP program, the outlined circles are functions/operators, and the outlined squares refer to as the terminals.

```
//r[1] = r[1] / r[1];
//r[3] = cf[0] + cf[5];
//if(r[3] < 0.86539)
//r[3] = r[3] - r[1];
r[0] = 0.453012 - cf[1];
//r[3] = r[2] * cf[5];
r[1] = r[0] * 0.89811;
if(cf[6] < cf[1])
r[2] = 0.453012 - cf[3];
r[3] = cf[4] - 0.86539;
```

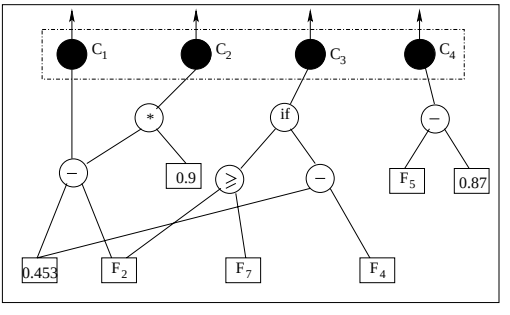


Fig. 4. An example evolved LGP program

As mentioned earlier, the eight feature terminals (`f1...f8`) correspond to the source registers (`cf[0]...cf[7]`), and the destination register (`r[0]...r[3]`) to the four class labels (`C1, C2, C3, C4`). Given an object to be classified, the destination register values can be easily calculated and the class of the object can be simply determined by taking the register with the largest value. For example, given the following four objects with different feature values,

	cf[0]	cf[1]	cf[2]	cf[3]	cf[4]	cf[5]	cf[6]	cf[7]
Obj1 (class1):	0.3056	0.3458	0.2917	0.2796	0.3052	0.1754	0.5432	0.5422
Obj2 (class2):	0.6449	0.6239	0.6452	0.6423	0.6682	0.7075	0.1716	0.1009
Obj3 (class3):	0.2783	0.3194	0.2784	0.2770	0.2383	0.2331	0.2349	0.0958
Obj4 (class4):	0.8238	0.7910	0.8176	0.8198	0.8666	0.8689	0.2410	0.1021

we can obtain the following target register values for each object example and make classification for each object image.

Object	Target-Class	r[0]	r[1]	r[2]	r[3]	Classified-Class
Obj1	class1	**0.1474**	0.13240	0.0000	-0.5602	class1
Obj2	class2	-0.1919	**-0.1723**	-0.1893	-0.1972	class2
Obj3	class3	0.1747	0.1569	**0.1760**	-0.6271	class3
Obj4	class4	-0.3708	-0.3330	-0.3668	**0.0012**	class4

According to the results, this program correctly classified all the four object examples. Other evolved programs have a similar pattern to this program. This suggests that the evolved LGP programs are relatively easier to interpret.

Further inspection of this program reveals that only four features were selected from the terminal set and some random numbers were also successfully evolved. This suggests that the LGP approach can automatically select features relevant to a particular task. The graph representation of the program also shows that the LGP approach can co-evolve sub-programs together each for a particular class and that some terminals and functions can be reused and/or shared by different sub-programs. This also suggests that the evolved LGP programs are more flexible than the highly constrained, fully inter-connected feed forward neural networks.

On the other hand, a TGP program can only produce a single value, which must be translated into a set of class labels but such a translation is often sensitive to the class boundaries. In other words, interpretation of a TGP program for image recognition/classification has to involves some kinds of additional program translation rule, which is a relatively indirect and difficult task.

4.3 Impact of Extra "Target" Registers

In TGP, Koza originally used *automatically defined functions* (ADFs) to evolve "subroutine" structures in GP. In such a design, the genetic programs and ADFs are evolved together and the programs can take the ADFs as functions or terminals. Koza [7] suggested that this approach would make better performance and more comprehensible programs. In LGP for multi-class image recognition programs, one way to simulate the ADFs in TGP is to use *extra target registers* in addition to the same number of target registers as the number of classes. In this sub section, we investigate the impact of the use of extra target registers, which serve as the ADFs, to the system performance.

The results on the *shape* data set using 1–10 extra target registers and same parameter values as in earlier experiments are shown in table 2. According to this table, the use of extra target registers as ADFs in LGP does not improve the system recognition accuracy at all. While the use of some extra (e.g. 3, 4, 6) registers resulted in similar performance, most extra registers led to clearly worse performance. The results on other data sets also showed a similar pattern, suggesting that the use of extra target registers (ADFs) in LGP does not improve system performance as claimed in TGP.

We believe the major reason is as follows. Firstly, while subprograms in ordinary TGP are very hard to be reused, the use of ADFs in TGP can clearly improve this situation. In LGP, however, due to the linear property, the programs are actually represented by a graph, where different parts of subprograms

Table 2. Impact of extra target registers on the shape data set

Extra registers	Test recognition rate (%)	Extra registers	Test recognition rate (%)
0	**99.84 ± 0.80**		
1	99.65 ± 1.74	6	99.36 ±2.66
2	98.68 ± 6.09	7	96.98 ± 8.66
3	99.31 ± 3.56	8	98.89 ± 4.94
4	99.29 ± 4.10	9	97.97 ± 6.32
5	98.14 ± 6.05	10	95.42 ± 9.37

can be easily reused. The program presented in Figure 4 shows such an example. In other words, LGP does not need extra target registers (ADFs) for subroutines and such a structure has already been embedded within its graph structure. Secondly, the use of extra target registers increases the size of the program space, which requires more effort to evolve good programs.

4.4 Impact of Program Length

In both LGP and TGP, there are some existing heuristics for setting program lengths [2] based on problem difficulty. However, there has not been any clear guidance for setting program lengths in LGP for multi-class image recognition problems. This sub section investigate this topic by varying the program length parameter ranging from *one fewer instructions than the number of classes* to *10 times as many instructions as the number of classes* for the three data sets. Due to the page limitation, only the results on the *shape* data set is shown in table 3.

Table 3. Impact of program length on the shape data set

Program length	Test recognition rate (%)	Program length	Test recognition rate (%)
3	89.46 ± 11.76	12	98.89 ±4.14
4	88.89 ± 11.64	**16**	**99.73 ±0.96**
5	98.16 ± 5.83	20	99.55 ± 2.61
6	97.33 ± 6.42	25	99.31 ± 3.63
8	97.98 ± 6.33	30	98.67 ± 1.30
10	98.85 ± 5.20	40	97.42 ± 3.53

As can be seen from the table, the LGP system with a too small program length cannot achieve good performance. As the this length increases to about *four times as many instructions as the number of classes*, the system obtained the best recognition accuracy. If this length continued to increase, the recognition accuracy will not increase (decrease a bit) but the evolution time got increased (time results are not shown here due to space limitation). The results on the other two data sets showed a similar pattern to the shape set. This suggests that when using LGP for multi-class recognition, a too small or a too large program length will not lead to the best performance, that there exists *a certain point* (more accurately *a certain range*) which can maximise the system performance,

and that the heuristic *four times as many instructions as the number of classes* can serve as a *starting point* for setting the program length parameter.

5 Conclusions

This paper investigated an LGP approach for multi-class image recognition and classification problems. This approach was compared with the basic TGP approach on three image data sets providing image recognition problems of increasing difficulty. The results suggest that the LGP approach outperformed the TGP approach on all the tasks in terms of recognition accuracy.

Inspection of the evolved genetic programs reveals that the program classifiers evolved by the LGP approach are relatively easy to interpret and that the LGP approach can co-evolve sub-programs together for multi-class image recognition.

The use of extra target registers as ADFs in LGP does not seem to improve the system performance for these problems. When using LGP for multi-class recognition, a too small or too large program length will not results in the best performance, and the heuristic *four times as many instructions as the number of classes* can serve as a starting point for setting the program length parameter.

Although developed for image recognition problems, we expect that this approach can be applied to general multi-class classification problems.

In future work, the relative strength of LGP on other more complex multi-class classification problems will be investigated and compared to commonly used classification methods such as neural networks and support vector machines.

References

1. Koza, J.R.: Genetic Programming. MIT Press, Campridge, Massachusetts (1992)
2. Banzhaf, W., Nordin, P., Keller, R.E., Francone, F.D.: Genetic Programming – An Introduction; On the Automatic Evolution of Computer Programs and its Applications. Morgan Kaufmann (1998)
3. Eggermont, J., Eiben, A.E., van Hemert, J.I.: A comparison of genetic programming variants for data classification. In: Proceedings of the Third Symposium on Intelligent Dat a Analysis (IDA-99), LNCS 1642, Springer-Verlag (1999)
4. Howard, D., Roberts, S.C., Ryan, C.: The boru data crawler for object detection tasks in machine vision. In Cagnoni, S. et al. eds.: Applications of Evolutionary Computing. Volume 2279 of LNCS., Springer-Verlag (2002) 220–230
5. Olaguea, G., Cagnoni, S., Lutton, E.: (eds.) special issue on evolutionary computer vision and image understanding, pattern recognition letters.**27**(11) (2006)
6. Krawiec, K., Bhanu, B.: Visual Learning by Coevolutionary Feature Synthesis. IEEE Trans. System, Man, and Cybernetics – Part B. **35** (2005) 409–425
7. Koza, J.R.: Genetic Programming II: Automatic Discovery of Reusable P rograms. MIT Press (1994)
8. Song, A., Loveard, T., Ciesielski:, V.: Towards genetic programming for texture classification. In: Proceedings of the 14th Australian Joint Conference on Artificial Intelligence, Springer Verlag (2001) 461–472

9. Loveard, T., Ciesielski, V.: Representing classification problems in genetic programming. In: Proceedings of the Congress on Evolutionary Computation. Volume 2., IEEE Press (2001) 1070–1077
10. Tackett, W.A.: Recombination, Selection, and the Genetic Construction of Computer Programs. PhD thesis, Faculty of the Graduate School, University of Southern California, Canoga Park, California, USA (1994)
11. Zhang, M., Ciesielski, V.: Genetic programming for multiple class object detection. In Foo, N., ed.: Proceedings of the 12th Australian Joint Conference on Artificial Intelligence (AI'99), LNAI Volume 1747. Springer (1999) 180–192
12. Zhang, M., Ciesielski, V., Andreae, P.: A domain independent window-approach to multiclass object detection using genetic programming. EURASIP Journal on Signal Processing, **2003** (2003) 841–859
13. Zhang, M., Smart, W.: Multiclass object classification using genetic programming. In Raidl, G.R. et al. eds.: Applications of Evolutionary Computing, Volume 3005 of LNCS., Springer Verlag (2004) 369–378
14. Fogelberg, C., Zhang, M.: Vuwlgp — an ANSI C++ linear genetic programming package. Technical report TR-CS-05-08, School of Mathematics, Statistics and Computer Science, Victoria University of Wellington (2005)
15. Oltean, M., Grosan, C., Oltean, M.: Encoding multiple solutions in a linear genetic programming chromosome. In Bubak, M., et al. eds.: Computational Science - ICCS 2004, Part III. Volume 3038 of LNCS. Springer(2004) 1281–1288
16. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning internal representations by error propagation. In Rumelhart, D.E., McClelland, J.L., the PDP research group, eds.: Parallel distributed Processing, Explorations in the Microstructure of Cognition, Volume 1: Foundations. Chapter 8. The MIT Press (1986)
17. Brameier, M., Banzhaf, W.: A comparison of genetic programming and neural networks in medical data analysis. Reihe CI 43/98, SFB 531, Dortmund University, Germany (1998)
18. Goldberg, D.E.: Genetic Algorithms in Search, Optimization, and Machine Learning. Addison–Wesley, Reading, MA (1989)
19. Georghiades, A., Belhumeur, P., Kriegman, D.: From few to many: Illumination cone models for face recognition under variable lighting and pose. IEEE Trans. Pattern Anal. Mach. Intelligence **23** (2001) 643–660

Evolving Texture Features by Genetic Programming

Melanie Aurnhammer

Sony Computer Science Laboratory Paris
6, rue Amyot, 75005 Paris, France
melanie@csl.sony.fr

Abstract. Feature extraction is a crucial step for Computer Vision applications. Finding appropriate features for an application often means hand-crafting task specific features with many parameters to tune. A generalisation to other applications or scenarios is in many cases not possible. Instead of engineering features, we describe an approach which uses Genetic Programming to generate features automatically. In addition, we do not predefine the dimension of the feature vector but pursue an iterative approach to generate an appropriate number of features. We present this approach on the problem of texture classification based on co-occurrence matrices. Our results are compared to those obtained by using seven Haralick texture features, as well as results reported in the literature on the same database. Our approach yielded a classification performance of up to 87% which is an improvement of 30% over the Haralick features. We achieved an improvement of 12% over previously reported results while reducing the dimension of the feature vector from 78 to four.

1 Introduction

Object recognition, image analysis, retrieval or classification and many other important tasks in Computer Vision rely on the selection of relevant features. The performance of these applications depends to a great extent on the suitability of the features chosen. Although feature selection is a very active research topic, no general solution for finding "the right features" has been found yet, for various reasons. Complex Computer Vision problems are often tackled by hand-crafting features, designed for the task at hand, but difficult if not impossible to generalise. Manually selected features are usually based on assumptions the researcher makes about similarities between objects or images of the same type. However, it cannot be guaranteed that these assumptions reflect the true characteristics of the object. Another well-known problem of feature selection concerns the dimension of the feature vector. The number of features can easily become very large. In order to find the "right" features from all available ones, computationally expensive component analysis methods, such as PCA or ICA, are commonly used.

In this paper, we investigate an alternative approach to generate features. Our work is based on the hypothesis that instead of engineering, features can

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 351–358, 2007.
© Springer-Verlag Berlin Heidelberg 2007

be evolved on a set of training examples by using a Genetic Programming [1] approach. The programmes are assembled from simple components, typically used for designing features. The quality of a feature is assessed on how well it can discriminate between the different classes. The feature vector is generated in an iterative process, where one feature is evolved in every run of the Genetic Programming method. Since the fitness of new features is calculated in combination with the features selected in previous iterations, we believe that new features will automatically be complementary to the existing ones. Furthermore, this procedure eliminates the need to later reduce the number of features by component analysis.

To verify this approach, we chose as testbed the application of texture classification based on co-occurrence matrices. Our choice was made considering: (1) the problem is well researched and free databases and publications are available. (2) The problem can be solved without relying on the results of prior segmentation. (3) The texture features developped by Haralick [2] are a well-accepted standard while being relatively simple and thus well-suited for comparisons. Furthermore, these features can deliver the basic components from which the Genetic Programmes are assembled.

2 Related Work

2.1 Texure Analysis

Texture analysis is a well researched area having produced a large number of different approaches. Perhaps the best-known method for statistical texture analysis was proposed by Haralick [2]. He originally derived fourteen features from "grey-tone spatial-dependance matrices", which are commonly referred to as grey-level co-occurrence matrices. Every entry (i,j) in a co-occurrence matrix describes the frequency of occurrence of pixels with grey values (i) and (j) between pixels separated by distance δ in a particular orientation α. Common values are $\delta = 1$, and $\alpha = 0°, 45°, 90°, 135°$. Although there exist many, more complex approaches for texture analysis, they often do not yield a significant improvement over the relatively simple Haralick features. For these reasons, Haralick's co-occurrence based features have been employed by many researchers. The seven most popular of these features are contrast, entropy, homogenity, angular second moment, correlation, inverse difference moment, and maximum probability.

2.2 Genetic Programming

Genetic Programming [1] is an approach to find solutions to a problem by automatically developing computer programmes. A population of such programmes is usually randomly assembled by problem specific components called terminals and functions. Then, Darwinian principles of natural selection and recombination are applied to evolve the population of programmes towards an effective solution to the given problem. Genetic Programming (GP) has been applied successfully

to a range of applications, including image enhancement and filtering (e.g. [3]), image analysis (e.g. [4]), object detection (e.g. [5]) and classification tasks (e.g. [6]). A recent work by Lam et Ciesielski [7] adresses an approach to employ GP for texture classification. Their approach differs from ours in the terminal and function sets used, i.e. they use the number of pixels at each grey level as terminals, and only the + function as function set. Furthermore, one feature for every combination of two classes is learned, which results in a high dimensional feature vector (78 dimensions for 13 classes).

3 A GP-Based Classification Framework

3.1 Method

Our approach to texture classification uses the same basic components as the Haralick features. It is based on the idea that GP will find better adapted features for a problem than the pre-defined Haralick features.

In our GP framework, a population of features is generated and evolved over a fixed number of iterations using selection, recombination and mutation mechanisms. A feature is a programme in form of a tree structure which is randomly assembled from *terminals* and *functions*. The *fitness* of a programme or feature is assessed by evaluating it on a set of training data and estimating its ability to discriminate between classes (see 3.2). When the evolution is terminated, the best feature over all generations is kept. The algorithm proceeds in an iterative manner to generate a vector of features. In the next iteration(s), the fitness function is evaluated for every combination of the best feature vector from past iteration(s) and all new features from the current iteration. This is based on the assumption that only complementary features will obtain high fitness values and are thus more likely to be selected for reproduction. Different criteria for terminating the procedure are possible: (1) after a fixed number of iterations has been reached, (2) when no significant improvement can be achieved by adding more features, or (3) when a satisfactory classification rate has been achieved.

3.2 Genetic Programming Components

The first step in designing a GP framework is to decide about the *function* and *terminal* set. The terminals for GP can usually be considered the data on which the functions are applied. Functions have a certain *arity*, which specifies the number of arguments or terminals. Programmes are randomly assembled from terminals and functions and their quality is calculated by a *fitness function*.

Terminals. In accordance with the Haralick texture features, we base our features on the co-occurrence matrices, i.e. choose them as terminals. We calculate the matrices for the four typical orientations $0°, 45°, 90°, 135°$ and use the full range of grey-values, which results in matrices of dimension 256 x 256. A constant distance of one pixel is used in our experiments since we found that strategies of increasing and randomly setting the distance did not yield better results.

Table 1. Function set. A and B denote the terminals, N the matrix dimension.

Functions					
Name	Arity	Definition	Name	Arity	Definition
$+$	2	$A+B=[a_{ij}+b_{ij}]$	mean	1	$\frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}a_{ij}$
$-$	2	$A-B=[a_{ij}-b_{ij}]$	var	1	$\frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}(a_{ij}-\text{mean})^2$
$*$	2	$A*B=[a_{ij}*b_{ij}]$	max	1	$\max[a_{ij}]$
$/$	2	$\begin{cases}[a_{ij}/b_{ij}] & \text{if } b_{ij}\neq 0,\\ 1 & \text{otherwise}\end{cases}$	abs	1	$\lvert a_{ij}\rvert$
$\sqrt{\ }$	1	$\sqrt{a_{ij}}$	ln	1	$\ln[a_{ij}]$
$\sum$	1	$\sum_{i=1}^{N}\sum_{j=1}^{N}a_{ij}$	pow2	1	$[a_{ij}]^2$

Functions. Our function set consists of simple mathematical operations as shown in Table 1. Some of the operations, such as the $\sum$ or *mean* function return a scalar instead of a matrix. This might constitute a problem for following operations, since results replace their parents in the tree structure. A common solution to this problem is to use strongly typed programming [8], where different operations are performed depending on the data type of the input. In our implementation, the operations *max*, $\sum$, *var*, and *mean* return the value itself in case the input terminal is already a scalar.

Fitness Function. The fitness of a feature is not only based on its classification performance but also on its discrimination abilities. A good feature will yield a good discrimination between classes, while it will show a small variation within classes. The *Fisher's discriminant ratio* (FDR) is a measure for the class separability which is defined as the ratio of the between-class scatter matrix S_b to the within-class scatter matrix S_w.

Let $\boldsymbol{x}_i$ be a set of N column vectors. The mean of the dataset is: $\boldsymbol{\mu}_x = \frac{1}{N}\sum_{i=1}^{N}\boldsymbol{x}_i$. There are K classes $\{C_1, C_2, \ldots, C_K\}$ and the mean of class k containing N_k members is: $\boldsymbol{\mu}_{xk} = \frac{1}{N_k}\sum_{\boldsymbol{x}_i\in C_k}\boldsymbol{x}_i$. The between class scatter matrix is then given by: $S_b = \sum_{k=1}^{K} N_k(\boldsymbol{\mu}_{xk}-\boldsymbol{\mu}_x)(\boldsymbol{\mu}_{xk}-\boldsymbol{\mu}_x)^T$. The within class scatter matrix is: $S_w = \sum_{k=1}^{K}\sum_{\boldsymbol{x}_i\in C_k}(\boldsymbol{x}_i-\boldsymbol{\mu}_{xk})(\boldsymbol{x}_i-\boldsymbol{\mu}_{xk})^T$. Combining Sb and Sw, the FDR results from $\text{FDR} = \frac{\text{trace}\{S_b\}}{\text{trace}\{S_w\}}$.

The classification performance is evaluated on the training set by using a minumum distance classifier. We assess the classification performance of a feature by the *F-score*, the harmonic mean of precision and recall:

$$\text{F-score} = \frac{2\times\text{recall}\times\text{precision}}{\text{recall}+\text{precision}}. \tag{1}$$

The fitness f_i of a feature i is then given by a combination of its discrimination ability and its classification performance on the training data: $f_i = \text{FDR} + 10\times$ F-score. The factor of 10 is used to give FDR and F-score a similar influence.

Classification. After the evolution finished, the best feature of all runs is selected and evaluated on the test data. A test sample is assigned to a class, by

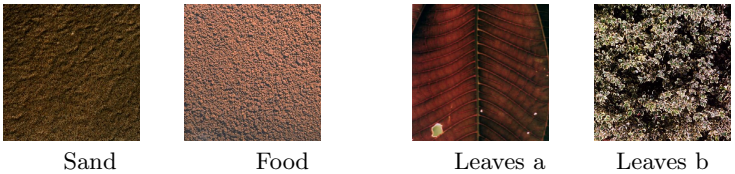
a simple minimum distance classifier. The classification performance is again evaluated by Equation 1.

3.3 Implementation

We implemented our texture classification system in C++, by using the OpenBeagle framework [9] for Genetic Programming as well as the Intel OpenCV library [10] for image processing. We used the standard Open Beagle evolver which includes the *ramped half-and-half initialisation* [1], a tournament selection mechanism, standard crossover and three different mutation operators. All mutation rates were set to 0.05, the crossover rate was 0.9. We used a population size of 150 for each GP run and chose a fixed number of 100 iterations as stopping criterion and restricted the maximum tree depth to 12.

4 Experiments

We evaluated our method on the publicly available database VisTex [11]. VisTex is a collection of common real-world photography samples which do not conform to rigid frontal plane perspectives and studio lighting conditions. VisTex contains a large number of classes (16) with a very small number of samples for some of the classes. Like other researchers [12], we find that VisTex class distributions are overlapping and most classes show a significant variability over their samples. Figure 1 highlights some of these difficulties.



Sand Food Leaves a Leaves b

(a) Strong overlap between classes (b) High interclass variability

Fig. 1. Examples from VisTex highlighting some difficulties of the database

We chose two different experimental settings to compare our method to approaches from other researchers. For the first setting, we follow [7] who took only one image per class and subdivided it into patches of 64x64 pixels. Half of these patches are used as training data, the other half as test data. We compare our results to those published by [7] as well as to results obtained by using the seven most popular Haralick features (see 2.1). In order to obtain a fair comparison, we tried to find excactly the same examples as in [7] which was achieved in all but one case. Like the authors, we used 15 classes. Our training and test database both contain thus 480 images. Figure 2 shows the classes we used from the VisTex database. Note that many of the images show perspective transforms, pattern irregularities, difficult lighting conditions, e.g. shadows, which make the

classification challenging. The second experimental setting uses a more realistic scenario, where we use all available images per class from VisTex, excluding only those classes with less than 6 images. This leaves ten classes for our experiments. In order to increase the number of available samples, we divided each image into four. The database was divided into 224 training images and 224 test images.

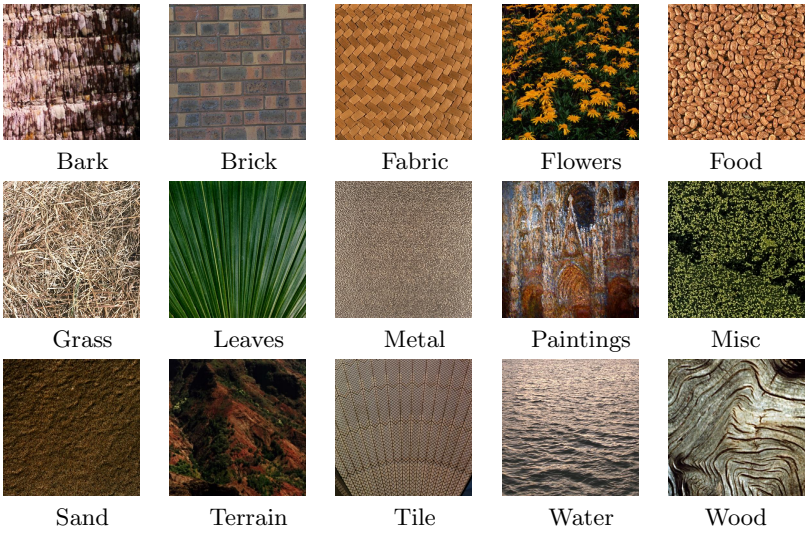


Fig. 2. The images from VisTex chosen for the first experimental setting

5 Results

The results obtained for the first setting are given in Table 2. It shows a comparision between the classification performance of the Haralick texture features (7-D) and a 4-D feature vector iteratively evolved by our GP approach. The results for the Haralick features were obtained with co-occurrence matrices for 90° which gave the best results. While the Haralick features achieved an overall classification rate of 67%, our approach yielded a classification rate of 87% using only four features. The classification rates were calculated by Equation 1. It can be seen that the GP-approach never obtains results below 0.5 (Paintings). For the Haralick features, a very low performance for the classes Bark (0.11), Paintings (0.27) and Terrain (0.39) was obtained. Only for one class (Brick), the Haralick features yielded a slightly better performance than the GP-approach (1.0 compared to 0.97). In [7], a classification accuracy of 74.8% is reported for a very similar experimental setting but using a feature vector of 78 dimensions.

The F-Score after the first iteration, i.e. for only one feature, is already 56%. Adding new features, it can be further increased to 87%. For a feature vector of more than four dimensions, no further improvement could be achieved.

Table 2. Classification Results: Haralick vs. GP features. One image per class.

	F-score			*F-score*	
Name	Haralick	GP Approach	Name	Haralick	GP Approach
Bark	0.11	0.60	Terrain	0.39	0.84
Brick	1.0	0.97	Food	0.70	0.93
Fabric	0.74	0.98	Grass	0.79	0.95
Flowers	0.52	0.81	Misc	0.86	1.0
Leaves	0.54	0.90	Tile	0.66	0.95
Metal	0.88	0.96	Water	0.61	0.94
Paintings	0.27	0.50	Wood	0.47	0.56
Sand	0.93	1.0			
total				0.67	0.87

The results of the second experimental setting are shown in Table 3. Since this problem is much more difficult, the performance is significantly lower than for setting one. The Haralick features yielded an overall performance of 47%, while our approach achieved a performance of 62% with a 5-D feature vector, an improvement of 32%. Using only one feature, the classification rate of our approach was already 47%.

Table 3. Classification Results. Haralick vs. GP features.

	F-score			*F-score*	
Name	Haralick	GP Approach	Name	Haralick	GP Approach
Bark	0.09	0.25	Metal	0.56	0.67
Brick	0.22	0.11	Paintings	0.46	0.52
Fabric	0.37	0.60	Sand	0.53	0.41
Flowers	0.57	0.40	Terrain	0.71	0.91
Leaves	0.32	0.68	Water	0.65	0.70
total				0.47	0.62

6 Conclusions

Our results support that GP is a powerful tool to automatically develop features. For our first experimental setting, we were able to achieve a classification rate 16% higher than reported in [7] with a very low dimensional feature vector (4-D), compared to 78-D. Further, our approach yielded better results than the hand-crafted Haralick features: 30% and 32% improvement for the first and second settings, respectively. This shows, that by evolving the features by GP, a better adaptation to the problem at hand can be achieved, although the same basic components are used. Our framework provides the possibility to develop feature vectors automatically without any user intervention and without prior specification of the number of features. By taking an iterative approach, we were able to develop a very low dimensional feature vector without the need for later

feature reduction by component analysis. The framework can easily be adapted to other databases or different problems.

Acknowledgements

The author would like to thank Luc Steels, François Pachet, and Pierre Roy for discussions, as well as Peter Hanappe, Martin Loetzsch, and Rafael Mayoral for their help with implementation issues.

References

1. Koza, J.R.: Genetic Programming: On the Programming of Computers by Means of Natural Selection. MIT Press, Cambridge, MA, USA (1992)
2. Haralick, R., Shanmugan, R., Dinstein, I.: Textural features for image classification. IEEE Transactions on Systems, Man, and Cybernetics **3**(6) (1973) 610–621
3. Poli, R., Cagnoni, S.: Genetic programming with user-driven selection: Experiments on the evolution of algorithms for image enhancement. In Koza, J.R., Deb, K., Dorigo, M., Fogel, D.B., Garzon, M., Iba, H., Riolo, R.L., eds.: Genetic Programming 1997: Proc. of the 2nd Annual Conference. (1997) 269–277
4. Poli, R.: Genetic programming for image analysis. Technical Report CSRP-96-1, University of Birmingham, UK (1996)
5. Winkeler, J.F., Manjunath, B.S.: Genetic programming for object detection. In Koza, J.R., Deb, K., Dorigo, M., Fogel, D.B., Garzon, M., Iba, H., Riolo, R.L., eds.: Genetic Programming 1997: Proceedings of the Second Annual Conference, Stanford University, CA, USA, Morgan Kaufmann (1997) 330–335
6. Smart, W., Zhang, M.: Classification strategies for image classification in genetic programming. In: Image and Vision Computing. (2003) 402–407
7. Lam, B., Ciesielski, V.: Discovery of human-competitive image texture feature extraction programs using genetic programming. In: GECCO (2). Volume 3103 of Lecture Notes in Computer Science. (2004) 1114–1125
8. Montana, D.J.: Strongly typed genetic programming. Technical Report #7866, Cambridge, MA 02138, USA (1993)
9. Gagné, C., Parizeau, M.: Open beagle: a generic C++ framework for evolutionary computation. `http://beagle.gel.ulaval.ca/index.html` (1999–2005)
10. Intel: Open source computer vision library. `http://www.intel.com/technology/computing/opencv/index.htm`(2005)
11. MIT: Vistex database. `http://vismod.media.mit.edu/vismod/imagery/VisionTexture/vistex.html` (1995)
12. Singh, S., Sharma, M.: Texture analysis experiments with meastex and vistex benchmarks. In Singh, S., Murshed, N., Kropatsch, W., eds.: Proc. Int. Conference on Advances in Pattern Recognition. Volume 2013 of LNCS., Springer (2001)

Euclidean Distance Fit of Ellipses with a Genetic Algorithm

Luis Gerardo de la Fraga, Israel Vite Silva, and Nareli Cruz-Cortés

Cinvestav. Department of Computing.
Av. Instituto Politécnico Nacional 2508. 07300 México, D.F., México
fraga@cs.cinvestav.mx

Abstract. We use a genetic algorithm to solve the problem, widely treated in the specialized literature, of fitting an ellipse to a set of given points. Our proposal uses as the objective function the minimization of the sum of orthogonal Euclidean distances from the given points to the curve; this is a non-linear problem which is usually solved using the minimization of the quadratic distances that allows to use the gradient and the numerical methods based on it, such as Gauss-Newton. The novelty of the proposed approach is that as we are using a GA, our algorithm does not need initialization, and uses the Euclidean distance as the objective function. We will also show that in our experiments, we are able to obtain better results than those previously reported. Additionally our solutions have a very low variance, which indicates the robustness of our approach.

Keywords: Ellipse Fitting, Euclidean Distance Fit, Genetic Algorithm.

1 Introduction

The problem of fitting a set of points to an ellipse has been treated intensively in the specialized literature since more than 200 years ago [1,2,3]. It is an important problem in Computer Vision, where the projection of a circle is an ellipse and it is necessary to recognize that ellipse.

The ellipse fitting problem has two solutions, named as *algebraic fit* and *geometric fit* [1]. An ellipse can be represented implicitly by the conic general equation, $ax^2 + bxy + cy^2 + dx + ey + f = 0$, and this equation represents an ellipse if $b^2 - 4ac < 0$. When we substitute the value of a point (x, y) in the ellipse equation, the value obtained from the implicit equation is called *algebraic distance*, and this formulation allows the most efficient algorithms because the fitting problem is linear and can be solved in a deterministic manner with a program that solves the generalized eigenvectors problem [2,3]. The main disadvantage of using an algebraic fitting algorithm is that the fitted ellipse will be distorted if the point values are not known with enough accuracy [1].

The best fit is obtained by using a geometric algorithm, because it takes into account the real Euclidean distance between a point and the ellipse. Now, we have a non linear problem that can be solved if it is linearized calculating the

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 359–366, 2007.
© Springer-Verlag Berlin Heidelberg 2007

Taylor series expansion of the Euclidean distance equation, and if we have an initial solution near to the real solution. In this case, it is necessary to iterate, and we need the derivative's direction in order to find the desired solution (the minimum). However, the Euclidean distance uses a square root, and its derivative is not continuous in all the search space. If the squared of the Euclidean distance, the quadratic distance, is used, then we can calculate a derivative that is continuous in all the search space and this approximation results in an iterative algorithm such as Gauss-Newton, steepest descent, etc. [4]. The main disadvantage of use the quadratic distance is that the farthest points will have more weight (the squares) than the nearest points to the curve. Other important disadvantage of these numerical methods is that it is necessary an initial solution near to the optimum solution to avoid getting trapped in local optima.

In this work we used a Genetic Algorithm (GA) to solve the optimization problem of fitting an ellipse to a set of given points, subject to finding the minimum of the sum of the Euclidean distances. With this point of view, the GA is a non-conventional (heuristic) method to solve an optimization problem, as they are the simulated annealing, an exhaust search or a random search. GAs have been successfully applied on a wide variety of engineering problems [5].

In this work we show that using a GA for ellipse fitting results in the following: (1) a better algorithm than the reported in [1]. It is better in the sense that produces a lower error (this error is measured as the sum of the Euclidean distances), (2) it is robust, in the sense that it always gives a good result , and (3) it solves problems of ellipse fitting with constraints that can not be solved in [1].

Of course, the disadvantages of using a GA is that the algorithm takes a greater execution time compared with the traditional numerical methods, and it gives a result of stochastic nature, i.e. we can not guarantee that the algorithm will always converge to the global optimum. However, we got solutions with a very low variance value. The advantages of use a GA is that we do not required an initial solution, and its result can be used to initialize a conventional numerical method. Furthermore, we can use Euclidean distances instead of quadratic distances.

This article is organized as follow, in Sec. 2 is presented the formal definition of the problem, in Sec. 3 are presented the results of ellipse fitting with and without constraints; finally, in Sec. 4 the conclusions of this work are presented.

2 Problem Definition

The Euclidean distance, d, from a point $\mathbf{x}$, to another point $\mathbf{x}'$, in $\mathcal{R}^2$, is represented by the equation:

$$d = \sqrt{(x_1 - x'_1)^2 + (x_2 - x'_2)^2} \tag{1}$$

The squared distance is d^2.

Our work is based on the algorithm *least-squares orthogonal distances fitting* (LSODF) of ellipses presented in [1]; we use the same part to calculate the

orthogonal contacting point on the ellipse, as it will be briefly explained next. The ellipse geometric equation is:

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1 \tag{2}$$

where a and b are the semimajor and semiminor axes, respectively. Eq. (2) can be rewritten in implicit form as:

$$f_1(x, y) = \frac{1}{2}(a^2y^2 + b^2x^2 - a^2b^2) = 0 \tag{3}$$

The constant $\frac{1}{2}$ in Eq. (3) simplifies the expressions for partial derivatives of f_1 respect to a and b.

A point $\mathbf{X}$ is on the ellipse, but the ellipse can be rotated and translated, then in order to apply Eq. (2), we need to calculate the points $\mathbf{x}$ using the relation $\mathbf{x} = \mathbf{R}(\mathbf{X} - \mathbf{X}_c)$, where $\mathbf{X}_c$ is the ellipse's origin, and

$$\mathbf{R} = \begin{bmatrix} \cos\alpha & \sin\alpha \\ -\sin\alpha & \cos\alpha \end{bmatrix}$$

where α is the pose angle defined as the positive angle between the semimajor and x-axis.

The transformed points are $\mathbf{x}_i$ with coordinates (x_i, y_i), $1 \leq i \leq N$; and the model expresed by Eq. (2) will be fitted to them. Now we need to calculate the point $\mathbf{x}'$ on the ellipse, and this is the nearest point to the point to be fitted. Point $\mathbf{x}_i$ must satisfy the equation:

$$f_2(x, y) = b^2x(y_i - y) - a^2y(x_i - x) = 0 \tag{4}$$

that represents the ellipse's normal in the point $\mathbf{x}'$.

The point $\mathbf{x}'$ is calculated by solving the non linear equation system formed by Eqs. (3) and (4). In order to do this, we used, as in [1], the Newton generalized method that converges in only three or four iterations.

The algorithm in [1] uses the Gauss-Newton method to estimate ellipse's parameters a, b, X_c, Y_c, and α that minimize the sum of quadratic distances.

Now, in our algorithm we use a simple Genetic Algorithm to estimate the ellipse's parameters, that minimize the error E. This error is calculated as the sum of the Euclidean distances:

$$E = \sum_{i=1}^{N} d_i \tag{5}$$

for N points and d defined in Eq. (1).

3 Ellipse Fitting

We experiment by applying a GA with two problems of ellipse fitting proposed in [1]: one without constraints, and other with two constraints. We also implemented in Octave [6] the algorithm in [1] to be able to calculate the sum of the Euclidean distances.

For all the GA's experiments the parameters values are the following: population size: 100; number of generations: 1,500; crossover type: one point; crossover rate: 0.8; selection type: stochastic remainder; representation: binary with Gray codes; mutation rate: $1/L$ where L is the chromosome's length in bits.

3.1 Ellipse Fitting Without Constraints

For the first experiment, we used the same 8 points as in [1]: $(1,7)$, $(2,6)$, $(5,8)$, $(7,7)$, $(9,5)$, $(3,7)$, $(6,2)$, $(8,4)$. In Table 1 we present the ellipse fitted as in [1]. The five design variables for this problem are: the axis lengths a and b, the ellipse's origin (X_c, Y_c), the pose angle α. The selected bounds for each variable are: $1 \leq a, b \leq 15$, $-10 \leq X_c, Y_c \leq 10$, and $0 \leq \alpha \leq \pi$.

All the variables are real numbers, encoded using binary representation with Gray codes. The required precision is 13 decimal places. The variables a and b need 47 bits each, x_c and y_c require 48 bits each, and α needs 44 bits, then the total chromosome's length is equal to 234 bits.

The fitness value for each individual in the population is assigned using Eq. (5).

Table 1. Ellipse fitted using LSODF algorithm [1]. We calculated the sum of quadratic distances and the sum of Euclidean distances.

Variable	a	b	x_c	y_c	α	$\sum d^2$	$\sum d$
Value	6.51872	3.03189	2.69962	3.81596	0.35962	1.37340	2.72777

In Table 2 we present the results of 50 runs of our GA, in this case we use 3,000 generations for all runs. The mean error, measured using Eq. (5), is lower, with a confidence interval of 95%, than the error obtained in [1], as can be seen in Table 1. Confidence intervals were measured using the bootstrap method. We consider this a difficult problem because the given points cover only partially the ellipse (see Fig. 1), and there could be several ellipses through those points with a small error value. This can also be seen in the variance of a and X_c in Table 2. Fig. 2 shows the variables values in one execution of the GA. Variables reach a steady value quickly. For this reason, we argue that our algorithm can be used to initialize, with few generations, a geometric fitting algorithm. In Fig. 3 is show the test of our algorithm with other sets of points; we ran 30 times the GA and the result that corresponds to the mean error is shown.

3.2 Ellipse Fitting with Constraints

Other experiment is to fit the set of points (it taken from [1]) $(8,1)$, $(3,6)$, $(2,3)$, $(7,7)$, $(6,1)$, $(6,10)$, $(4,0)$, to an ellipse with two constraints: $\alpha = \pi/2$, and $ab = 20$. Two variables are eliminated of the GA, α and b (b is calculated as $20/a$). Our algorithm is able to reach these constraints, in contrast to the algorithm in [1] that can not give a solution that satisfies the constraint $\alpha = \pi/2$, and of course the error is very large as can be seen in Table 3. The design variables for

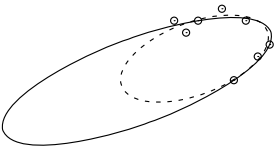


Fig. 1. Results of [1] (dashed line), and GA (solid line); it is shown the ellipse corresponding to the mean error

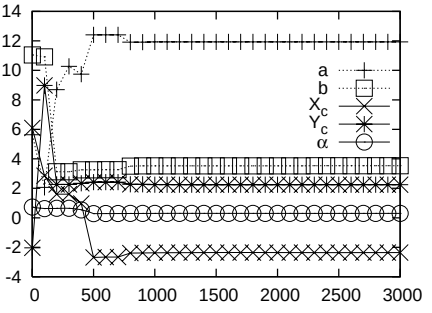


Fig. 2. One example of the variables' values that fit to one ellipse without constraints, vs. the number of generations. Values were taken every 100 generations.

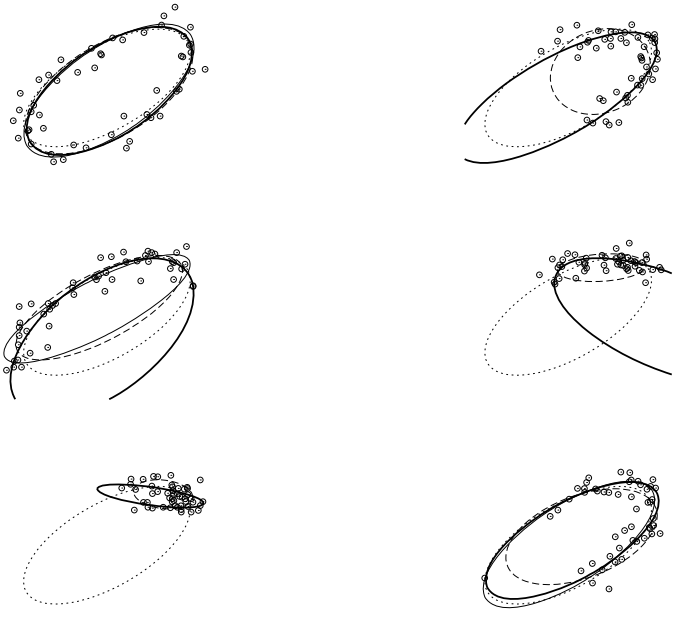


Fig. 3. We used an ellipse of axes 4.5 and 2, with center at $(5, 4)$ and rotated 30° (dotted line); then we generate 50 points and to each point position a Gaussian noise was added to obtain a SNR of 10. We calculate three ellipses: an algebraic fitted ellipse (dashed line), using the LSODF algorithm [1] (solid line) initialized with the previous ellipse, and our GA fit (bold line). In some cases the LSODF algorithm does not converged. The selected bounds were $0 \leq a, b, X_c, Y_c \leq 10$, and $0 \leq \alpha \leq \pi$.

this problem are three, with bounds: $1 \leq a \leq 20$, and $-10 \leq X_c, Y_c \leq 10$. The error measured with Eq. (5) is lower than the obtained by algorithm in [1] by the obvious reason that it does not satisfy the constraints. Fig. 4 shows graphically

Table 2. Results of 50 runs of our algorithm for fitting an ellipse without constraints. Ellipse's values best and worst correspond to the best and worst error values. Mean values correspond to the algorithm output with the error nearest to the average error. s.d. is the standard deviation for the 50 corresponding column values; c.i. confidence interval for the mean at confidence level of 95%.

Var.	a	b	X_c	Y_c	α	$\sum d$
Best	13.90334	3.79800	-4.17695	1.49402	0.31664	2.31152
Mean	11.96368	3.57339	-2.12886	1.91188	0.35554	2.40696
Worst	6.33286	2.93675	2.83207	3.95779	0.25020	2.51333
s.d.	2.34601	0.27181	2.12739	0.84695	0.03781	0.04855
c.i.	[10.10,11.50]	[3.375,3.528]	[-1.805,-0.508]	[2.188,2.693]	[0.305,0.327]	[2.396,2.424]

Table 3. Ellipse fitted with the algorithm in [1]. The constraint are $\alpha = \pi/2$, and $b = 20/a$. This result does not satisfy the second constraint. We calculate both, the sum of quadratic distances, and the sum of Euclidean distances.

Variable	a	b	X_c	Y_c	$\sum d^2$	$\sum d$
Value	4.58061	4.36740	6.17691	4.46768	358.04831	49.79220

Table 4. Results of 50 runs of our GA for fitting an ellipse with two constraints: $\alpha = \pi/2$, and $b = 20/a$. See explanation for all values in caption of Table 2.

Variable	a	b	X_c	Y_c	$\sum d$
Best	8.57001	2.33371	5.07927	2.12460	3.93865
Mean	8.59337	2.32737	5.06726	2.12613	3.96133
Worst	8.44411	2.36851	5.03589	2.28686	4.00552
s.d.	0.03912	0.01076	0.01368	0.03854	0.01716
c.i.	[8.526,8.550]	[2.340,2.346]	[5.072,5.080]	[2.146,2.169]	[3.956,3.966]

the ellipse obtained by both algorithms, and Fig. 5 shows the variables values with respect to the number of generations for one execution of the GA.

3.3 Analysis of Parameters

In order to know how sensitive is the algorithm to its parameters' values we conducted an Analysis of Variance (ANOVA).

We selected four parameters to be studied, they are: *Population Size, Selection Type, Mutation Rate, Crossover Type.* They are called *independent variables.* The *dependent variable* is the fitness value.

The selected parameter values (levels) assigned to each independent variable are the following:

- Population Size (three levels): **50**, **100** and **200** individuals.
- Selection Type (three levels): **Binary Tournament**, **Roulette** and **Stochastic Remainder**.

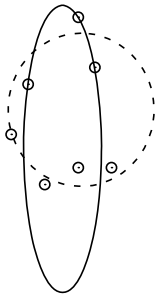


Fig. 4. Result of [1] (dashed line) and our GA (continuous line); it is shown the ellipse corresponding to the mean error

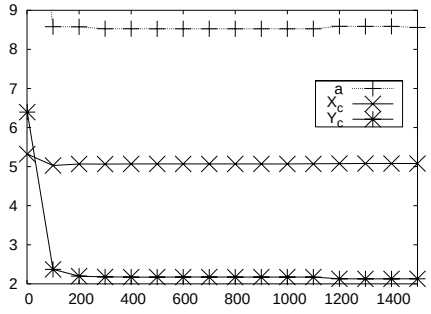


Fig. 5. One example of the fitted variables' values for the ellipse with two constraints: $\alpha = \pi/2$ and $ab = 20$, vs. the number of generations. Values were taken every 100 generations.

- Mutation Rate (three levels): **0.1**, **1/L** and **2/L**, where **L** = chromosome's length.
- Crossover Type (two levels): **One Point** and **Two Points**.

The *Number of Generations* was established such that the number of fitness function evaluations is the same for each experiment (1,000,000 in our case). We performed 30 independent GA runs with each possible combination of parameter levels (54 possibilities). From this ANOVA we can set the following conclusions:

– The *Selection* type applied to the algorithm has a real effect on the algorithm's performance. The probability that its effect is produced by the randomness is less than 0.001. The best results were obtained when using *Roulette* or *Stochastic Remainder* (proportional techniques). The worst results were obtained when using *Binary Tournament*.

– The value assigned to the parameters *Crossover type* and *Population Size* has not real effect on the algorithms' performance.

– The value assigned to the *Mutation Rate* parameter has a real effect on the algorithm's performance. The probability that its effect is due to the randomness is less than 0.001. The best results were obtained when applying Mutation Rate equal to 0.1

4 Conclusions

We presented a GA to solve the problem of fitting an ellipse to a set of points, taking as the objective function the sum of the Euclidean distance from the given points to the curve. This problem can not be solved by a conventional numerical method (Newton or Gauss) because the derivative of the Euclidean distance is not continuous in all the search space.

The advantages to use a GA are: we do not need to initialize our proposed algorithm with a solution near to the global optimum, as it is required by a conventional numerical method; in all the executions we got a reasonable solution; the GA gives better results than the algorithm [1], that uses the Gauss-Newton method, according to the error measured as the sum of the Euclidean distances; we consider the GA robust because it is not trapped in local minima.

The main disadvantage of a GA is that the execution time is high. For one run, 3000 generations and population size of 100, with 50 data points, the execution time was 84 sec, for 8 points it was 49 sec. Execution time was measured in a iBook G4 with a 1.07 GHz processor, and with Mac OS X version 10.3.9. We think that the binary representation used is not the best one, because we are using floating point variables; therefore could be better uses other evolutionary technique that uses directly real numbers such as the Differential Evolution.

In fact, the number of generations can be reduced if the GA, or other evolutionary technique, is adapted specifically to solve this problem. We are working on that.

Our proposed algorithm can also be used to initialize a conventional numerical algorithm, i.e. the result at 500 generations can be used for that.

Acknowledgments

This work was partially supported by CONACyT, México under grant 45306.

References

1. S.J. Ahn, W. Rauth, and H-J. Warnecke. Least-squares orthogonal distances fitting of circle, sphere, ellipse, hyperbola, and parabola. *Pattern Recognition*, 34(12):2283–2303, Dec 2001.
2. A. Fitzgibbon, M. Pilu, and R.B. Fisher. Direct least square fitting of ellipses. *IEEE Patt. An. & Mach. Intell.*, 21(5), May 1999.
3. P. O'Leary and P. Zsombor-Murray. Direct and specific least-square fitting of hiperbolae and ellipses. *Jounal of Electronic Imaging*, 13(3):492–503, Jul 2004.
4. K. Deb. *Optimization for Engineering Design*. Prentice-Hall, 2002.
5. R. Leardi. Genetic algorithms in chemometrics and chemistry: a review. *Journal of Chemometrics*, 15(7):559–569, 2001.
6. Gnu octave, a high-level language for numerical computations. www.octave.org.

A Particle Swarm Optimizer Applied to Soft Morphological Filters for Periodic Noise Reduction

T.Y. Ji, Z. Lu, and Q.H. Wu*,**

Department of Electrical Engineering and Electronics
The University of Liverpool, Liverpool, L69 3GJ, U.K.
q.h.wu@liverpool.ac.uk

Abstract. The removal of periodic noise is an important problem in image processing. To avoid using the time-consuming methods that require Fourier transform, a simple and efficient spatial filter based on soft mathematical morphology (MM) is proposed in this paper. The soft morphological filter (Soft MF) is optimized by an improved particle swarm optimizer with passive congregation (PSOPC) subject to the least mean square error criterion. The performance of this new filter and its comparison with other commonly used filters are also analyzed, which shows that it is more effective in reducing both periodic and non-periodic noise meanwhile preserving the details of the original image.

Keywords: Soft Morphological Filter, Particle Swarm Optimizer, Periodic Noise.

1 Introduction

Periodic noise is a kind of noise that widely exists in raw images due to electrical interference from data collecting devices. Removal or reduction of the periodic noise is a fundamental problem in image processing.

Because periodic noise has a well-defined frequency, a usual approach is to eliminate the noise in the frequency domain. However, despite all their advantages, frequency filters are always computationally time-consuming, due to the conversion between the space and frequency domains and the noise peak detecting procedure.

MM is an effective nonlinear approach to image processing in the space domain. Based on set theory and containing simple arithmetical operations, MM filter spends less computational time than other traditional filters and is easy to design. In this paper, Soft MF, a kind of morphological filter based on soft morphology is advanced.

Evolving MM algorithms using a genetic programming approach has been extensively studied in [1], which aims to extract a particular feature of a binary image. In this paper, an optimization technique using PSOPC is applied

* Corresponding author: Tel: +44 1517944535, Fax: +44 1517944540.
** Senior Member IEEE.

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 367–374, 2007.
© Springer-Verlag Berlin Heidelberg 2007

to design Soft MF so as to achieve better filtering results for gray scale images. Particle swarm optimizer (PSO) has shown a faster convergence rate than other evolutionary algorithms on some problems and it has very few parameters to adjust [2]. Moreover, by introducing passive congregation, the information sharing mechanism is improved and the optimization result is more accurate [3].

2 Soft MF for Periodic Noise Removal

Soft MM is an extension of standard MM. Instead of applying local maximum and minimum operations, soft MM uses a more general weighted order statistics [4]. Besides, the structuring element (SE) used in soft MM is divided into two subsets: the hard center and the soft boundary. Compared with standard MM, soft MM is more robust in noisy conditions and less sensitive to small variations in the shapes of the objects [5].

Given sets $A, B \subseteq Z^2$ and $A \subseteq B$, B is divided into two subsets: the hard center A and the soft boundary $B - A$. Soft dilation and erosion of an image f by the SE $[B, A, k]$ are defined as:

$$f \oplus [B, A, k] = \max^{(k)}\{k \diamond (f(x-\alpha)+A(\alpha)) | \alpha \in D_A\} \bigcup \{f(x-\beta)+B(\beta) | \beta \in D_{B-A}\}, \tag{1}$$

$$f \ominus [B, A, k] = \min^{(k)}\{k \diamond (f(x+\alpha)-A(\alpha)) | \alpha \in D_A\} \bigcup \{f(x+\beta)-B(\beta) | \beta \in D_{B-A}\}, \tag{2}$$

where $\max^{(k)}$ and $\min^{(k)}$ denote the k^{th} largest and smallest value in the set respectively; $\diamond$ is the repetition operator and $k \diamond f(a) = \{f(a), f(a) \cdot \cdot, f(a)\}$ (k times); D_A, D_{B-A} represent the field of definition of A and $B - A$ respectively.

A simple Soft MF is designed as calculating the average of soft dilation and soft erosion:

$$f_{\text{softm}} = (f \oplus [B, A, 2] + f \ominus [B, A, 2])/2, \tag{3}$$

with a $5 * 5$ SE, the middle one as shown in equation (9). This filter can suppress both the positive and negative parts of the noise while preserve the signal impulses. In this paper, Soft MF depicted by equation (3) is denoted as *Average Soft MF*.

In this paper, a typical periodic noise - sinusoidal noise is added to the original image. Sinusoidal noise is a kind of commonly appeared periodic noise, denoted as $N(\omega, \theta, \alpha)$, where ω, θ and α represent the frequency, angle and amplitude respectively. ω means that the noise is with a period length of $1/\omega$ pixels. θ varies from 0 to 180. α reflects the changes in gray value caused by noise and $\alpha = 0$ means no noise is added to the image. Figure 1(b) shows the result of adding sinusoidal noise $N(1/4, 30, 30)$ to the original image Figure 1(a).

3 Optimization of Soft MFs Using PSOPC

Average Soft MF described above is only a simple choice. A more general formula of Soft MF should be

$$f_{\text{softm}} = \alpha \cdot (f \oplus [B, A, k]) + \beta \cdot (f \ominus [B, A, k]), \tag{4}$$



(a) Image Pepper (b) Contaminated by noise $N(1/4, 30, 30)$

Fig. 1. Add periodic noise to the original image

where $0 < \alpha < 1$ and $0 < \beta < 1$ suffice $\alpha + \beta = 1$. Since the value of α, β, k and the design of B, A influence the filter's performance directly, it is believed that better results could be achieved if the parameters are optimized.

3.1 PSOPC

PSO is a population based algorithm developed in 1995 [2][6], which shares many similarities with other iteration based evolutionary computation techniques: usually initialize the system with a group of randomly generated population, evaluate fitness values to update the population, and search for the optimum solutions by updating generations, the strategy of which is based on the previous generations. The updating algorithm of PSO is: in every iteration, each particle is updated by two best values: (a) the best solution it has achieved so far, called personal best (*pbest*); (b) the best solution achieved by any particle in the population in this iteration. If the best solution is among all the particles, it is called global best (*gbest*); if it is taken from some smaller number of adjacent particles, it is called local best (*lbest*). After finding the two best values, the particle updates its velocity and positions according to the following equations [2][6]:

$$V_i^{k+1} = wV_i^k + c_1 r_1 (P_i^k - X_i^k) + c_2 r_2 (P_g^k - X_i^k), \tag{5}$$

$$X_i^{k+1} = X_i^k + V_i^{k+1}, \tag{6}$$

where V_i^k is the velocity of the i^{th} particle in the k^{th} iteration; X_i is the position of the i^{th} particle; P_i is the *pbest* position of the i^{th} particle; P_g is the *gbest* or *lbest* position; r_1, r_2 are two random numbers between $(0, 1)$; w, c_1, c_2 are learning factors, usually being set to $w = 0.75$, $c_1 = c_2 = 2.05$ [7]. It should also be noted that particles' velocities on each dimension are restricted within a predefined range $[0, V_{\max}]$. If the velocity tends to exceed this range, it is limited to $V_{\max}$.

PSOPC is a new PSO with passive congregation, firstly proposed in [3]. In passive congregation, group members can get necessary information from not

only the environment but also their neighbors [3][8]. Therefore, individuals in the swarm have more options to obtain information, which helps to minimize the chance of missed detection and incorrect interpretations. PSOPC is defined as [3]:

$$V_i^{k+1} = wV_i^k + c_1 r_1(P_i^k - X_i^k) + c_2 r_2(P_g^k - X_i^k) + c_3 r_3(R_i^k - X_i^k), \quad (7)$$

where R_i is a particle randomly selected from the swarm as the neighbor of the current individual; c_3 is the passive congregation coefficient and is set to $c_3 = 0.6$ [3]; r_3 is a uniform random sequence in the range $(0, 1) : r_3 \sim U(0, 1)$. It has been proved in [3] that PSOPC finds better results than standard PSO, and for most of the unimodal functions, PSOPC is more accurate and converges faster.

3.2 Optimization of Soft MF

As stated in equation (4), in order to design an optimal soft MF, the following parameters should be considered:

1. The size of the SE;
2. The shape of the hard center, also the shape of the soft boundary;
3. The repetition operator k;
4. The choice of the weight coefficients α and $\beta = 1 - \alpha$.

Since the larger the SE is, the blurrier the output will be, the size of SE is limited to $3 * 3$ and $5 * 5$. Therefore, considering symmetry, the hard center of SE can only be chosen from the following:

$$\begin{bmatrix} 0\,1\,0 \\ 1\,1\,1 \\ 0\,1\,0 \end{bmatrix} \text{ or } \begin{bmatrix} 0\,0\,0 \\ 0\,1\,0 \\ 0\,0\,0 \end{bmatrix}, \quad (8)$$

$$\begin{bmatrix} 0\,0\,1\,0\,0 \\ 0\,1\,1\,1\,0 \\ 1\,1\,1\,1\,1 \\ 0\,1\,1\,1\,0 \\ 0\,0\,1\,0\,0 \end{bmatrix} \text{ or } \begin{bmatrix} 0\,0\,0\,0\,0 \\ 0\,1\,1\,1\,0 \\ 0\,1\,1\,1\,0 \\ 0\,1\,1\,1\,0 \\ 0\,0\,0\,0\,0 \end{bmatrix} \text{ or } \begin{bmatrix} 0\,0\,1\,0\,0 \\ 0\,0\,1\,0\,0 \\ 1\,1\,1\,1\,1 \\ 0\,0\,1\,0\,0 \\ 0\,0\,1\,0\,0 \end{bmatrix}. \quad (9)$$

Accordingly, the repetition parameter is an integer between $[1, 3]$ or $[1, 5]$, subject to the size of the SE. α may vary from 0 to 1. The fitness function is chosen to calculate Mean Square Error, defined as:

$$MSE = \frac{1}{MN} \sum_{i=1}^{M} \sum_{j=1}^{N} [I_o(i, j) - I_f(i, j)]^2, \quad (10)$$

where MSE stands for Mean Square Error, $I_o(i, j)$ and $I_f(i, j)$ are the original image and the filtered (output) image respectively, M and N denote the dimensions of the image. Accordingly, Peak Signal Noise Ratio is defined as:

$$PSNR = 20 \log_{10}(255/\sqrt{MSE}), \quad (11)$$

and is employed to quantitatively evaluate the quality of the filtered image.

The optimization procedure is carried out in the following conditions: Image Pepper is selected as the original image, and is contaminated by sinusoidal noise $N(1/4, 30, 30)$. When optimizing an integer parameter, the corresponding value of the particle is equiprobably mapped to a valid integer. When evaluating the fitness function, the MSE is only calculated within a small part of the image in order to reduce the computational complexity. When initializing the system, the size of the swarm is chosen to be 30 and the maximum iteration is set to 300. It is testified by experiments that the fitness value remains at 419.8367 after iterating 300 times or after the size of population reaches 30. The optimization result is listed below:

1. The size of SE is $5 * 5$;
2. The hard center of the SE is the third one in equation (9);
3. The repetition operator is $k = 2$;
4. The weight coefficients are set to be $\alpha = 0.527$ and $\beta = 0.473$.

However, the optimizing results of α and β are not the same when different experiments are carried out in the same conditions. Applying the optimization procedure 10 times and the results vary slightly around 0.5, therefore, considering the adaptivity of the optimization algorithm, the weighted coefficients are set to be $\alpha = \beta = 0.5$.

3.3 Simulation Results and Analysis

Removal of Sinusoidal Noise. Figure 2 shows the results of removing sinusoidal noise $N(1/4, 30, 30)$ using four different filters: (a) Optimum Soft MF, (b) Average Soft MF, (c) Median Filter with a $5 * 5$ window and (d) Spectral Median Filter, an effective filter for periodic noise removal in the frequency domain recently proposed in [9]. It applies the technique used in the spatial domain median filter to spectrum amplitude image, and checks whether the amplitude of the pixel (i, j) (denoted as X_{ij}) is a peak. If X_{ij} suffices $\frac{X_{ij}}{MED_{m*n}(X_{ij})} \geq \Theta$, then it is a peak and should be replaced by $MED_{m*n}(X_{ij})$. Since it is not discussed in [9] how to determine the parameters, in this paper they are chosen by trial and are set to be $m = 9$, $n = 9$ and $\Theta = 2.2$.

Intuitively, the Median Filter rarely removes any noise. On the contrary, Spectral Median Filter oppresses the noise but blurs the image. As for the other two filters, Optimum Soft MF outperforms Average Soft MF greatly.

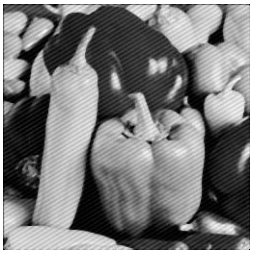
To quantitatively analyze their performance, the $PSNR$s of the output images are calculated. The comparison of the filtering results is shown in Figure 3(a), where the periodic length $1/\omega$ of the sinusoidal noise $N(\omega, 30, 30)$ varies from 2 to 12. It should be noticed that before filtering, the $PSNR$s are close to $20.9dB$. As can be seen from the figure, as the width of noise increases, the $PSNR$s decrease severely except Spectral Median Filter. It is mainly because the size of the SE used in Soft MFs and that of the window used in Median Filter are not large enough to cover the noise. On the contrary, the $PSNR$ of Spectral Median



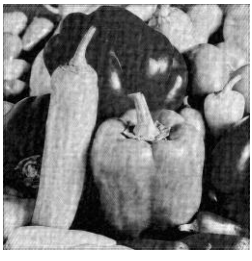
(a) Optimum Soft MF



(b) Average Soft MF

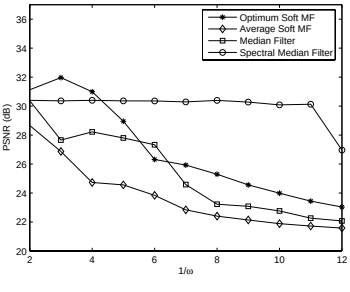


(c) Median Filter

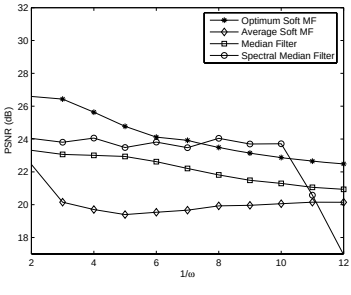


(d) Spectral Median Filter

Fig. 2. Removal of sinusoidal noise $N(1/4, 30, 30)$



(a) Sinusoidal noise

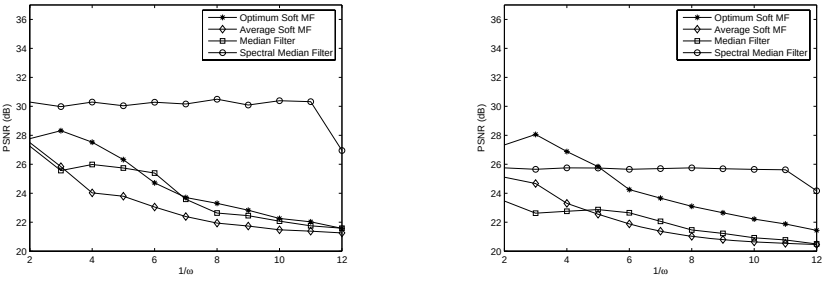


(b) Compound noise

Fig. 3. Comparison of $PSNR$s in noise removal of Image Pepper

Filter maintains at a high level thanks to a noise peak detecting procedure, which is time-consuming yet effective.

This Optimum Soft MF is also applied to two other noise corrupted images to evaluate its adaptability. The other three filters are also employed to compare their performance, and the results are illustrated in Figure 4. As can be seen, Optimum Soft MF outperforms Average Soft MF and Median Filter in



(a) Image Lena (b) Image Bridge

Fig. 4. Comparison of $PSNR$s in sinusoidal noise removal of Image Lena and Image Bridge

most cases, and when the frequency of the periodic noise is higher, it even outperforms Spectral Median Filter. But as the frequency decreases, the result of Optimum Soft MF gets worse while Spectral Median Filter is not that sensitive to frequency.

Removal of Compound Noise. In practical instances, periodic noise is usually companied with random noise, such as Gaussian white noise. Therefore, it is necessary to evaluate the filters' ability to reduce compound noise.

Corrupting Image Pepper by both periodic noise $N(1/\omega, 30, 30)$ and Gaussian white noise $mean = 0, variance = 0.01$, then filtering them by the four filters respectively, the simulation results are shown in Figure 3(b). As can be seen from the figures, when Gaussian white noise is added, the $PSNR$s of all the filters descend remarkably. To discuss them separately, Spectral Median Filter cannot remove Gaussian white noise effectively, while both the two spatial filters are able to suppress it to a great extent without extra computation.

Computation Efficiency. Although the overall performance of Optimum Soft MF is not as good as Spectral Median Filter in noise reduction, the computation time of the former is much less. The average computation time of Optimum Soft MF is 3 ms while that of Spectral Median Filter is 6.5 ms. All simulations are implemented in Matlab 7.0 running on a PC with a CPU of Intel Pentium IV 2.80 GHz.

4 Conclusion

This paper has discussed the performance of soft MFs in 2D periodic noise removal and has proposed an approach of optimizing the filter using PSOPC. As a simply combined soft MF, Average Soft MF is better at noise suppressing compared with Median Filter when the frequency of the noise is high and the implementation is not complicated. Nevertheless, it cannot be proved that Average Soft MF is the best approach of designing a filter. In order to enhance

the adaptability, an optimization technique using PSOPC to search the optimum soft MF has been advanced, which evaluates the size and the shape of the SE, the repetition parameter and the weight coefficients. By introducing passive congregation to PSO, particles are able to gain more information from their neighbors thus avoid the risk of misjudging the updating direction. Besides, only one extra coefficient is needed.

Among all the filters discussed in the paper, Optimum Soft MF shows the most satisfactory performance in periodic noise reduction and small shape preservation with less computation time. It outperforms Average Soft MF in that its $PSNR$ is enhanced without an evident increase in computation expense. Compared with Median Filter, it is outstanding in its significant improvement in detail preserving as well as in its ability to reduce the noise. Although Optimum Soft MF is not as good as Spectral Median Filter in noise reduction especially when the frequency of the noise is low, it is less computational time-consuming. Besides, the performance of Spectral Median Filter depends on the choice of parameters, yet the principles to determine them are lacked.

References

1. Quintana, M.I., Poli, R., Claridge, E.: Morphological algorithm design for binary images using genetic programming. Genetic Programming and Evolvable Machines **7**(1) (March 2006) 81–102
2. Kennedy, J., Eberhart, R.C.: Particle swarm optimization. In: Proceedings of IEEE International Conference on Neural Networks. Volume IV. IEEE Press, Piscataway, NJ (1995) 1942–1948
3. He, S., Wu, Q.H., Wen, J.Y., Saunders, J.R., Paton, R.C.: A particle swarm optimizer with passive congregation. Biosystems **78** (2004) 135–147
4. Gasteratos, A., Andreadis, I., Tsalides, P.: Fuzzy soft mathematical morphology. Vision, Image Signal Processing, IEE Proceedings **145**(1) (1988) 41–49
5. Hamid, M., Harvey, N., Marshall, S.: Genetic algorithm optimisation of multidimensional grey-scale soft morphological filters with applications in archive film restoration. Circuits and Systems for Video Technology, IEEE Transactions **13**(5) (May 2003) 406–416
6. Eberhart, R.C., Kennedy, J.: A new optimizer using particle swarm theory. In: Proceedings of the Sixth International Symposium on Micro Machine and Human Science. Kluwer Academic Publishers, Nagoya, Japan (1995) 39–43
7. Clerc, M., Kennedy, J.: The particle swarm - explosion, stability, and convergence in a multidimensional complex space. IEEE Transactions on Evolutionary Computation **6**(1) (February 2002) 58–73
8. Parrish, J.K., Hamner, W.M.: Animal groups in three dimensions. Cambridge University Press, Cambridge, UK (1997)
9. Aizenberg, I., Butakoff, C.: Frequency domain median-like filter fo periodic and quasi-periodic noise removal. In: SPIE Proceedings of Image Processing: Algorithms and Systems. (2002) 181–191

Fast Genetic Scan Matching Using Corresponding Point Measurements in Mobile Robotics

Kristijan Lenac[1,2], Enzo Mumolo[1], and Massimiliano Nolich[1]

[1] DEEI, University of Trieste, Via Valerio 10, Trieste, Italy
[2] AIBS Lab, Via del Follatoio 12, Trieste, Italy

Abstract. In this paper we address the problem of aligning two partially overlapping surfaces represented by points obtained in subsequent 2D scans for mobile robot pose estimation. The measured points representation contains incomplete measurements. We solve this problem by minimizing an alignment error via a genetic algorithm. Moreover, we propose an alignment metric based on a look-up table built during the first scan. Experimental results related to the convergence of the proposed algorithm are reported. We compare our approach with other scan matching algorithms proposed in the literature, and we show that our approach is faster and more accurate than the others.

1 Introduction

Scan matching for robot localization can be briefly described in the following way: given a 2D reference scan gathered with a range sensing device by the robot located in a given pose and another 2D scan gathered by the robot located in another pose, the goal is to determine the 2D rigid motion that maximizes the scan overlapping. The scan matching is performed by comparing the coordinates of corresponding scan points to be matched. The problem of identifying the points to be matched in different scans is a fundamental aspect of many matching algorithms.

The matching and analysis of geometric shapes is an important problem that arises in various applications areas, in particular computer vision, pattern recognition and robotics. Widely used heuristic methods for aligning 2D or 3D point sets are variations of the Iterative Closest Point algorithm (ICP) [1]. It has three basic steps: first, pair each point of the first set to the second one using a corresponding criterion; second, compute the rotation and translation transformations which minimize the mean square error between the paired points and finally apply the transformation to the first set. The optimum matching is obtained by iterating the three basic steps. However, ICP has several drawbacks. First, its proper convergence is not guaranteed, as the problem is characterized by local minima. Second, ICP requires a good prealignment of the views to converge to the best global solution.

Several variants of ICP have been proposed [2]. Another approach for aligning the two sets of points is to find the geometric transformation through a

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 375–382, 2007.
© Springer-Verlag Berlin Heidelberg 2007

pose-search, rather than the corresponding-search of ICP. The search space of geometric transformations contains solutions that can be used to align the two sets. In this case the objective is to find a solution close to the global optimum, in a reasonable time.

Most of the work in shape matching has been performed in the 3D case but the same approaches can be applied in the 2D case as well. Yamany et al. [3] used a genetic algorithm for registration of partially overlapping 2D and 3D data by minimizing the mean square error cost function. Silva et al. [4] used GAs for alignment of multiview range images. A novel Surface Interpenetration Measure is introduced that allows to evaluate registration results more precisely, especially in the case of small registration errors. A predefined residual threshold is used to handle partial overlap. However, no method is given to adjust the threshold automatically. Lomonosov et al. proposed in [5] a combined approach in which genetic search is used to initialize an efficient iterative alignment procedure. Their system is applicable to arbitrarily oriented surfaces while preserving precision and robustness of iterative methods.

In this paper a fast genetic look-up based algorithm for scan matching for mobile robots is presented. The main contributions of our approach are the following:

- it does not require corresponding point computation;
- it can be both fast and accurate properly selecting the grid used in look-up table;
- the same algorithm can be applied both in local scan matching and in global scan matching.

This paper is organized as follows. In Section 2 we describe the algorithm proposed in this paper. In Section 3 some experimental results are reported, including some comparisons with other algorithms, to show some advantage in accuracy and computational time in a practical case. In Section 4 some concluding remarks are reported.

2 Genetic Look-Up Based Algorithm for Scan Matching (GLASM)

The algorithm proposed in this paper will be called GLASM in the following. It aims at finding the (x,y) translation and the rotation Φ that allow to obtain the best alignment between two different scans. The algorithm is based on the computation of a lookup table which divides the plane of scan readings in a grid for a rough but fast reference point look-up as will be shown next in the description of the fitness function.

Each parameter of the scan position (x, y, Φ) is coded in the chromosome as a string of bits as follows: *nbitx* for coding x, *nbity* for y and *nbitrot* for Φ.

The problem space is discretized in a finite solution space with a resolution that depends on the extension of the search area and on the number of bits in each gene. The search area limits can be set based on the problem at hand. In

the case of pose tracking where odometry measurements are available they are usually set based on the odometry error model. The algorithm performs well even for very large search areas like in global scan matching where no initial position estimate is available to narrow the search space.

The positional information is coded in the genes using the Gray code. To obtain the position from the gene the inverse Gray code value of the bit string is taken. In this way the variations in the bit string caused by the mutation or crossover operators translate in the proportional variations in the position they represent. Using a typical binary code a change in one bit may cause a significant variation in the position. We found that for the scan matching application, typical binary code leads to a reduced efficiency of the genetic algorithm.

The genetic algorithm starts with a population of Popsize chromosomes which is randomly selected with uniform distribution in the search space. Each chromosome represents a single position of the new scan. The goal of the scan matching is to estimate the position of the new scan relative to a reference scan or a given map which is best fitted according to a fitness value.

2.1 Fitness Function

The fitness computation must be done very quickly and it must provide a value that closely correlates with the degree of matching between two scans.

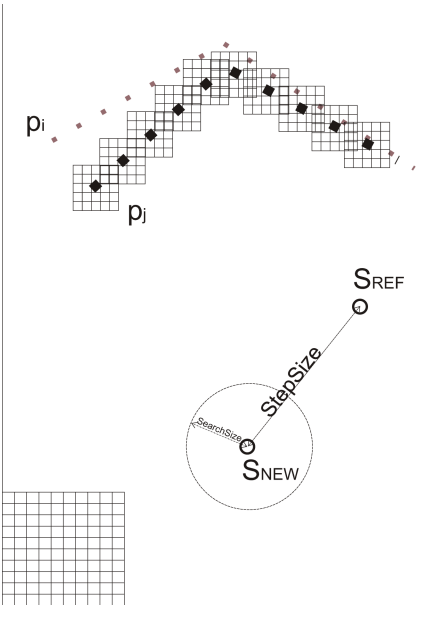


Fig. 1. The environment is discretized and a look-up table is built. The cells near the points of the reference scan are depicted. These cells are the only ones to be marked in the look-up table.

As soon as the reference scan is known, a look-up table is constructed as depicted in Fig. 1. Each cell represents a portion of the space of the 2D plane of scan readings. The overall dimensions of the lookup table cover all the search space. In the case of pose tracking the lookup table is centered in the reference scan position. In order to cover all the possible sensor readings of the new scan it is at least as large as the step size plus the sensor range. For position estimation with an a-priori given map the lookup table should at least cover the map. Each cell in the table is marked with a boolean value 0 or 1. The cells of the table are initialized with 0 and only the cells close to a reference scan point are marked with 1.

The genetic algorithm evaluates the fitness value for a chromosome in the following way: for each point of the new scan a direct lookup is made in the lookup table to establish if the point has a corresponding one or not, i.e. if the cell corresponding to the point has 1. The number of points of a new scan having a corresponding point in a reference scan is then taken as a fitness value for this chromosome, i.e. for the new scan evaluated in the position coded in a chromosome. This in fact is directly proportional to the overlapping of two scans, i.e. the degree of matching between the two.

In this way there is no need for the correspondence function since no pairings need to be done between scan points. There is also no need for the matching error function since the fitness value is directly obtained by counting positive lookup returns. The speed-up gain of our approach comes at the cost of building the initial lookup table and of a somewhat reduced accuracy of the matching process posed by quantization errors introduced by the lookup table. However the experiments presented in this paper show that both of them are negligible. In fact the creation of the initial lookup table is very fast and the accuracy reduction is lower or comparable to the resolution of the finite solution space, so the overall effect is negligible.

In order to avoid some areas carrying more points than others the scans are resampled. The objective function would otherwise favor the superposition of dense areas (with more points) which may lead to erroneous matching.

On the basis of the above considerations, it is clear that we define our basic fitness function as the sum of the squares around each point coming from one scan that intercepts the point in the new scan:

$$f_{S_{NEW},S_{REF}} = \sum_{i=1}^{N} \rho(i) \tag{1}$$

where S_{NEW} and S_{REF} are the two scans to be matched, N is the number of points in S_{NEW}, and the value of ρ is as follows:

$$\rho(i) = \begin{cases} 1, \text{ if a point of } S_{NEW} \text{ lies in the square around a point in } S_{REF} \\ 0, \qquad\qquad \text{otherwise} \end{cases} \tag{2}$$

2.2 Advantages and Limitations

Other genetic algorithm's fitness functions, as for example GCP [3] algorithm, first search for corresponding point pairs and then compute the mean of the

distances between them. To speedup the corresponding points search procedure, we instead directly evaluate the fitness by matching points which, after the projection on the same coordinate frame, fall in the search window around the previous scan. Hence, our approach does not yield a closest point but has a linear computational complexity $O(N)$, whereas GCP and ICP corresponding phases have a quadratic cost of $O(N^2)$.

It must be pointed out that the presented algorithm introduces some limitations to the overall accuracy of the matching process. These are given by the resolution of the finite solution space (which is set by the number of bits used to code a position in the genes and by the extension of the search space) and by a quantization of the fitness function which, as defined with a lookup table, may label different chromosomes with the same value if they represent positions which are very near to each other, so a certain grouping of chromosomes is possible depending on the size of the marked areas in the lookup table around points of the reference scan. In the worst case, two identical S_{REF} and S_{NEW} scans with no rotational error must be aligned. In this case all the positions in the area equivalent to the size of the marked squares give the same fitness value. So it is a good idea to take this size as small as possible. On the other hand it shouldn't be too small since non marked cells may arise in the space between adjacent readings in a reference scan. If the scan matching is performed with a known geometric map instead then this is not the case since the cells near the walls of the map may be marked without interruptions, i.e. the lookup table may be created by simply growing the map walls.

An appropriate size of the marked area may be chosen for the problem at hand depending on the resolution of the solution space and the angular resolution of the range sensor (or equivalently the resampling of the scans defining the distance between readings).

Let us now turn to the memory needed for the implementation of the look-up table. The look-up table is a 2D array requiring 1 bit of information per cell. For a scan matching in robotic applications typical sensor ranges are between 5 to 50 meters (sonar, laser scanner). With an exploration step size of several meters a cell size of 10 cm or less might be appropriate. To cover a map of 100x100 meters (which is surely more than enough considering that it is a local map centered in the reference scan) an array of 1 million cells is necessary which gives approximately only 125 kb of required memory.

It is important to note that the look-up table concept may be extended to accommodate for the corresponding function and matching error function typically required in classic correspondence search algorithms, i.e. the information identifying the nearby corresponding point of the reference scan may be stored in the marked cells and the resolution of the look-up table may be increased to directly provide a distance between paired points for the matching error function. In this case the memory usage goes up by an order of magnitude, nevertheless it is still easily implemented in a typical computer with few MB of memory.

3 Experimental Results

The experimental results have been obtained with the data set acquired with a focalized ultrasonic sensors described in [6] and the data resolution is about 20 cm, which is the spot size obtained with the focalizing device. Moreover, since the reflection depends on the rugosity of the surface, in some cases the reading from the sensor is missing. Let us describe now some experimental findings. In Fig. 2 some convergence behaviors of GLASM are reported considering

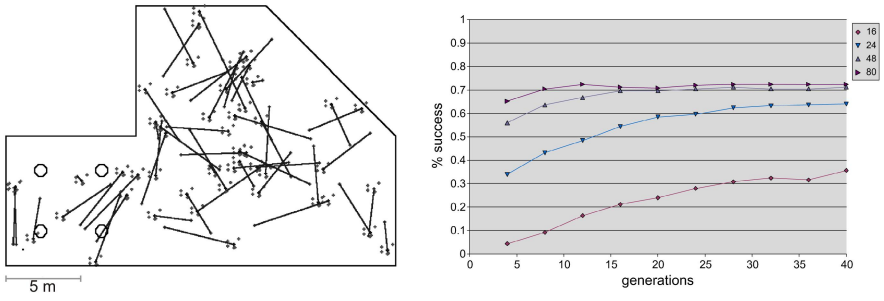


Fig. 2. (Left panel) Random scan pairs used in tests. For each pair 5 different initial position estimates were used. The search space is centered in the initial position estimate given by the odometry. The extension of the search space is of 0.8 m, 0.8 m, 0.8 rad for x, y, and Φ, respectively. (Right panel) Convergence behavior of GLASM in terms of success ratio versus generation number for different population sizes.

the dimension of a real environment. In the left panel the configuration of the environment is shown, together with an initial set of poses. It is important to note that the results reported in Fig. 2, in Tab. 1 and in Tab. 2 are expressed in terms of success ratio. We define success ratio as the ratio between the number of localization trials that give an estimated position within the ellipsoid centered in the true position (x_{true},y_{true},Φ_{true}). The radii of the ellipsoid in the implementation were 0.05 m on the x and y axis and 0.05 rad for the rotation, i.e. the matching was considered successful if the resulting position was at least 5 cm near the true position and rotated less than 0.05 rad from it. In other words, the success ratio represents the accuracy of the pose estimation procedure. Finally, it is important to note that all the experimental results reported in this section were obtained with a chromosome length of 18 bits, namely 6 bits for *nbitx*, 6 bits for *nbity*, and 6 bits for *nbitrot*.

For example, in Fig. 3 the populations at two different stages of convergence are shown.

GLASM has been compared to a pure 2D point-to-point version of the ICP algorithm (without tangent line calculations), as reported in [7]. As ICP requires a good pre-alignment of the scan pairs, to obtain comparable results the extension

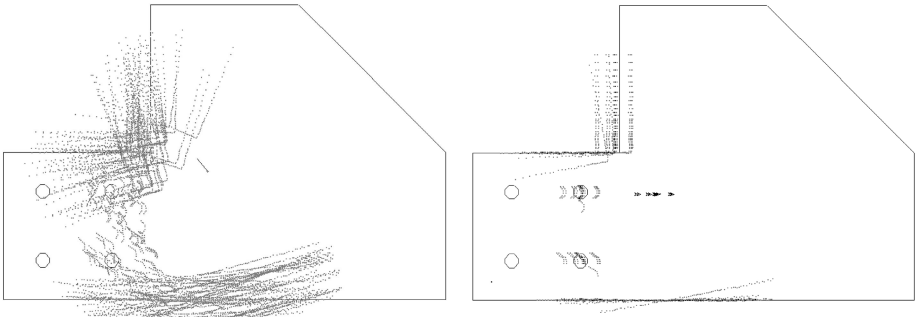


Fig. 3. Example of how GLASM works. In the left panel are shown the individuals of the population at an initial stage, in the right panel are shown the individuals of a population near the optimum.

Table 1. Comparison between Iterative Closest Point (ICP) algorithm and the proposed (GLASM) algorithm (Population size = 30; Number of generations = 10; Crossover probability = 1; Mutation probability = 0.0092). (MDE = Mean Distance Error in meter; MRE = Mean Rotation Error in meter; VDE = Variance of Distance Error; VRE = Variance of Rotation Error).

Algorithm	Mean Exec Time (ms)	Success Ratio	MDE	MRE	VDE	VRE
ICP	270	0.752	0.033124	0.011058	0.000646	0.000225
GLASM	85	0.952	0.023685	0.001733	0.000249	0.000010

Table 2. Comparison between PGA and GLASM (Population size = 60; Number of generations = 10; Crossover probability = 1; Mutation probability = 0.0092). (MDE = Mean Distance Error in meter; MRE = Mean Rotation Error in meter; VDE = Variance of Distance Error; VRE = Variance of Rotation Error).

Algorithm	Mean Exec Time (ms)	Success Ratio	MDE	MRE	VDE	VRE
PGA	728	0.800	0.021441	0.001875	0.000118	0.000012
GLASM	171	0.904	0.001659	0.001659	0.000114	0.000008

of the search space has been reduced to 0.4 m, 0.4 m, 0.4 rad for x, y, and Φ, respectively. In Tab. 1 some results are reported using the same set of 50 scan pairs and the same 5 robot positions, randomly selected. Using GLASM a faster satisfactory result has been obtained than using ICP.

GLASM has also been compared to the genetic algorithm based on a polar approach (Polar Genetic Algorithm, PGA) as described in [7]. In Tab. 2 some results are reported. The same set of 50 scan pairs and the same 5 robot positions have been considered. Furthermore, the same population size and number of generations have been used for both the algorithms. GLASM has a faster convergence to the optimal solution than PGA.

4 Final Remarks and Conclusions

In this paper we have dealt with 2D scan matching for mobile robot localization. A genetic algorithm based on a look-up table and Gray coding of the parameters for scan matching is proposed. Iterative and genetic optimization schemes have been compared, showing that the proposed GLASM using a look-up table is both fast and accurate.

To summarize the reported experimental results, the accuracy improvement with respect to classical solutions to the matching problem goes from about 13% to about 37% for the considered environment for the classical ICP and the polar GA scan matching algorithms respectively. In the same experimental conditions, the computational time is reduced by a factor ranging from 3 to 4 approximatively.

Current work is devoted to a real-time implementation of the algorithm for a mobile robot navigation task based on pose estimation.

References

1. P. Besl, N. McKay, *A method for registration of 3D shapes*, IEEE Transaction on Pattern Analysis and Machine Intelligence, V.14, pp.239–256, 1992.
2. S. Rusinkiewicz, M. Levoy, *Efficient Variants of the ICP Algorithm*, Proc. Int. Conf. on 3D Digital Imaging and Modeling", pp.145–152, 2001.
3. S. Yamany, M. Ahmed, A. Farag, *A new genetic based technique for matching 3D curves and surfaces*, Pattern Recognition, V.32, pp.1817–1820, 1999.
4. L. Silva, O. Bellon, K. Boyer, *Enhanced robust genetic algorithms for multiview range image registration*, Proc. Int. Conf. on 3D Digital Image and Modeling, pp.268–275, 2003.
5. E. Lomonosov, D. Chetverikov, A. Ekrt, *Pre-registration of arbitrarily oriented 3D surfaces using a genetic algorithm*, Pattern Recognition Letters, Special Issue on Evolutionary Computer Vision and Image Understanding, V.27, pp.1201–1208, 2006.
6. E. Mumolo, K. Lenac, M. Nolich, *Spatial Map Building using Fast Texture Analysis of Rotating Sonar Sensor Data for Mobile Robots*, International Journal of Pattern Recognition and Artificial Intelligence V.19/1, pp.1–20, 2005.
7. J. L. Martinez, J. Gonzalez, J. Morales, A. Mandow, A. J. Garcia-Cerezo, *Mobile Robot Motion Estimation by 2D Scan Matching with Genetic and Iterative Closest Point Algorithms*, Journal of Field Robotics, V.23/1, pp.21–34, 2006.

Overcompressing JPEG Images with Evolution Algorithms

Jacques Lévy Véhel[1], Franklin Mendivil[2], and Evelyne Lutton[1]

[1] Inria, Complex Team, 78153 Le Chesnay, France
jacques.levy-vehel@inria.fr, evelyne.lutton@inria.fr
[2] University of Acadia, Department of Mathematics and Statistics, Wolfville, Nova Scotia, Canada, B4P 2R6
franklin.mendivil@acadiau.ca>

Abstract. Overcompression is the process of post-processing compressed images to gain either further size reduction or improved quality. This is made possible by the fact that the set of all "reasonable" images has a sparse structure. In this work, we apply this idea to the overcompression of JPEG images: We reduce the blocking artifacts commonly seen in JPEG images by allowing the low frequency coefficients of the DCT to vary slightly. Evolutionary strategies are used in order to guide the modification of the coefficients towards a smoother image.

1 Statement of the Problem

Various compression methods have been devised in view of reducing the size of image files for purposes of storing and transmission. In order to reach substantial compression rates, lossy techniques have to be used, *i.e.* the compressed/decompressed image is a degraded version of the original one. Most such compression methods allow to tune the size of the compressed file, so as to reach a trade-off between reduction and quality.

Many people have realized the following fact: The set of all "reasonable" images is extremely small as compared to the one of all "possible" images. Although the term "reasonable" is vague, the meaning is clear: If one chooses at random the gray level values of all pixels in a $N \times N$ image, where, say, $N = 512$ and the gray levels are coded on 8 bits, then the probability that the result looks like a meaningful image is ridiculously small. One may wonder if it is possible to improve the efficiency of the various compression methods by using this remark. We term attempts of this kind *overcompression*. Overcompression is not a new compression method, nor is it specific to a given compression scheme. Rather, overcompression tries either to further reducing the size of the compressed file or to improving of the quality of the decoded image, by taking advantage of the sparse structure of the set of images.

Although overcompression is by no means an easy task, it may be approached by a variety of methods. In this work, we propose an overcompression scheme for the case of JPEG compressed images.

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 383–390, 2007.
© Springer-Verlag Berlin Heidelberg 2007

The JPEG compression format [1] is the most popular image compression method to date. It has served as a standard until recently. Although JPEG has now been surpassed by a new standard, called JPEG 2000 [1], it is still widely used for several reasons. A major one is that a huge number of images are stored in this format, and it does not seem feasible nor desirable to acquire again all these data in order to process them with the new format. One may also mention the fact that while JPEG is public domain, JPEG 2000 uses some patented techniques, which reduces its diffusion.

It thus seems desirable to increase the efficiency of JPEG. As explained below, our overcompression method improves on the quality of JPEG images by reducing the blocking artifacts classically encountered with this compression method.

Several methods have already been proposed in view of post-processing JPEG-compressed images in order to remove the blocking artifacts. See [2] for a comprehensive list of references.

2 Overcompressing JPEG Images

The general problem of restoring fidelity in a degraded image is almost impossibly difficult. However, since we are assuming that our image is a compressed JPEG image, we know how the information is lost. Our method is particularly adapted to the case of medium to high compression ratios, which translates into noticeable blocking effects.

The basic JPEG compression algorithm decomposes an image into 8×8 non-overlapping blocks and treats each such block independently of all others. The DCT (Discrete Cosine Transform) of each block is computed and then these frequency coefficients are quantized. Since the details of this quantization are important for our methods, we discuss them further below. After the quantization, the 8×8 table of coefficients is linearly ordered using a zig-zag traversal of the array, run length encoded and then finally some type of entropy coding is applied.

The 64 frequency coefficients are quantized individually by dividing by a *quantization value* and then rounding to the nearest integer. There are two important points to notice about this process. The first is that many different DCT values become quantized to the same integer value. The second is that these "equivalent" DCT values lie in an interval whose length is the size of the corresponding quantization value. Thus the higher this quantization value, the wider the corresponding interval (and the more information which is lost in the process).

Because of this information loss, many initial images all lead to the same final JPEG image (all their differences being removed by the quantization). We call this a *JPEG equivalence class* of images. Given a compressed JPEG image, clearly both the initial image and the JPEG image are in this equivalence class (as are many others).

Our basic idea is to move around within this equivalence class to remove the blocking artifacts. The fact that the original image lies within this class ensures that there are some images which are visually better than the compressed one.

Table 1. Example of *quantization values* for JPEG

16	11	10	16	24	40	51	61
12	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

Searching within an equivalence class mearly necessitates perturbing the given JPEG DCT coefficients in such a way that each value remains in the given interval. This is simple to ensure by constraining the size of the perturbations according to their frequency component.

It is important to mention that we assume that the JPEG is reasonably close to the image. We cannot recover lost information, we only try to smooth out the blocking artifacts at the block boundaries without oversmoothing the entire image.

Each block is represented in the DCT domain by a matrix of 8×8 coefficients, where the frequency of coefficient (i, j) increases in the x (resp. y direction) direction as i (resp. j) increases. Thus, low-frequency (resp. high-frequency) coefficients are the ones with low (resp. high) value of $i + j$. Since we are only interested in smoothing the blocking artifacts, and not in recovering the lost high-frequency content, we need only modify the low-frequency coefficients. Experiments showed that tuning the 6 coefficients corresponding to the lowest frequency components (*i.e.* the ones with $i + j \leq 4$ - recall that i and j takes values in $\{1, \ldots, 8\}$) allowed to reach our purpose. More precisely, we ran the following test: We replaced these coefficients in highly compressed JPEG images with the "correct" coefficients from the uncompressed images. This resulted in images which were almost indistinguishable from the non-compressed ones. As a consequence, correcting these coefficients should be sufficient to restore most of the image quality.

We thus need to optimize 6 parameters per 8×8 block. For a 256×256 image, this still means processing 6144 values. This is beyond the capacity of any reasonable algorithm. However, we can take advantage of the following remark: JPEG ignores inter-block dependence. As a matter of fact, this is precisely the reason why blocking artifacts arise. This can be exploited in a very simple way, resulting in an algorithm which can easily be parallelized (we did not try this explicitly): Our algorithms decompose the image into a collection of overlapping "tiles", which are processed independently. The results for all the tiles are then blended together by using a convex combination of the various tiles at all places where the tiles overlapped.

This results in huge savings in computational effort and thus much better solutions: For a 256×256 image, using 24×24 tiles and shifting by 8 pixels in

each direction results in 900 tiles, so 900 optimizations with 54 parameters each. This contrasts to a single optimization with 6144 parameters. The value 24 is justified by the fact that, with this size, each block is processed once along with its 8 neighbors, allowing an efficient treatment of the blocking effects.

The optimizations on each tile are very fast and result in good local solutions. The blended global solution was found to be much better than any of the solutions we obtained to the global problem, even with higher iteration counts.

Even though the solutions obtained this way are probably sub-optimal, the justification is that the necessary adjustment to the DCT coefficients in disjoint tiles are approximately independent, with this approximation becoming more true as the distance between the tiles increases.

3 Fitness Function

As always, the choice of the fitness function is of the utmost importance. This is particularly crucial in our case, because there is no obvious way to measure the adequacy of a solution.

Since the aim is to reduce the blocking artifacts of a highly compressed JPEG image, the fitness function should provide a quantitative measurement of these artifacts. There are many possible ways of trying to do this, and we discuss several that were tried, along with some which were tried but then discarded.

As previously mentioned, both the original and the JPEG image are in the given equivalence class. The fitness function should obviously yield a smaller (more optimal) value on the original image than on the JPEG image (since we wish for images which look more like the original image than the JPEG image). This simple criterion eliminates several candidate fitness functions. In particular, one might be tempted to use fitness functions that provide a measure of the smoothness of the image, with the idea that a highly compressed image will not be as smooth as the original image, due to the blocking artifacts. One such metric is the *total variation norm* [3], a commonly used metric in image processing. However, this metric is rather ill-adapted to our case: Indeed, a highly compressed JPEG image tends to be almost piecewise constant, because it consists of large almost flat regions bounded by a comparatively small number of edges between blocks. This is precisely the type of images that is favored by total variation. As a consequence, compressed images will often tend to have smaller fitness values than their original version.

The second type of fitness function is one which explicitly uses the 8×8 blocking structure of JPEG. There many possibilities for this as well. Among those tried were:

1. Summing the absolute differences across all the interior 8×8 block boundaries.
2. For each interface between two blocks, compute the average value on both sides of the edge and sum the absolute value of these differences.
3. Same as 2) except normalize by some block measure (like the mean).

4. For each interface between two blocks, compute the vector of differences along the edge and accumulate the L1 norm of these vector differences divided by the variance of the vector.

In each case, the idea is that one wishes to penalize large inter-block differences but allow for the fact that blocks may consist of texture regions.

4 Evolutionary Algorithms

To find optimal perturbation of the JPEG coefficients, we tested two algorithms: a $(1+1)$ EA and a genetic algorithm.

If we write the JPEG image as a sum over the blocks, $I = \sum_B I_B$, we can represent the individual as a sum of perturbations, $\delta = \sum_B \delta_B$. We are looking for the best δ, that is so that $I + \delta$ has smoothed out the blocking artifacts. Thus each $\delta_B \in \mathbb{R}^6$, with each component appropriately constrained (by the condition that $I_B + \delta_B$ still lie in the correct quantization interval).

Modifying a component consists of adding a random perturbation (uniformly distributed in an interval) to that component, wrapping around if the new value lies outside the particular quantization interval. We use wrap-around rather than clipping since the former ensures a uniform distribution over the interval (experiments showed that clipping yields lower quality results).

After testing various possibilities, the fitness function used was item 1) from the list above, simply summing the absolute differences across all (internal) block boundaries. Because only the lowest frequency coefficients were modified, the resulting images were not too irregular: Since we do not touch the coefficients (i, j) such that $i + j > 4$, we have no risk of introducing spurious higher frequencies, and there is no need to introduce a term measuring this in the fitness function.

4.1 $(1+1)$ EA

We tried a simple $(1+1)$ EA, that is, each generation consists of a single individual which produces one mutation which may become the next generation.

We mutate δ by independently modifying each component with some probability p_c, obtaining δ'.

If $f(\delta') < f(\delta)$, we always replace δ with δ'. On the other hand, with probability p_r we replace δ with δ' even if $f(\delta) < f(\delta')$.

4.2 Genetic Algorithm

A classical genetic algorithm was used, with the following parameters: uniform selection, elite count = 2, crossover fraction = 0.8, uniform mutation with probability = 0.1 and range two thirds of the quantization bin (*i.e.* the size of the interval where the uniform perturbation is drawn depends on the quantization bin, and is equal to two thirds of the length of this bin), population size = 40, number of generations = 200.

5 Experimental Results

We display in figures 1 to 3 results on two images: The well-known lena image, and an image of a tree. Both images are 256×256 with the grey levels coded on 8 bits.



Fig. 1. Compression results, compression factor 5



Fig. 2. Compression results, compression factor 10

We show optimization with both the $(1+1)$ EA and the genetic algorithm. Three compression ratios have been considered: The compressed images are obtained by using the quantization values in table 1 multiplied by 5, 10, and 15.



Fig. 3. Lena with compression factor 15 (left), optimized with the GA (right)

References

1. Home site of the JPEG and JBIG committees: http://www.jpeg.org/
2. Nosratinia, A. *Enhancement of JPEG-Compressed images by re-application of JPEG*, Journal of VLSI Signal Processing, **27**, (2001), 69-79.
3. Chan, T.F., Shen, J., Zhou, H-M. *Total Variation Wavelet Inpainting*, Journal of Mathematical Imaging and Vision, **25**-1, (2006), 107–125.

Towards Dynamic Fitness Based Partitioning for IntraVascular UltraSound Image Analysis

Rui Li[1], Jeroen Eggermont[2], Michael T.M. Emmerich[1],
Ernst G.P. Bovenkamp[2], Thomas Bäck[1],
Jouke Dijkstra[2], and Johan H.C. Reiber[2]

[1] Natural Computing Group, Leiden University,
P.O. Box 9512, 2300 RA Leiden, The Netherlands
{ruili,emmerich,baeck}@liacs.nl
[2] Division of Image Processing, Department of Radiology C2S,
Leiden University Medical Center,
P.O. Box 9600, 2300 RC Leiden, The Netherlands
{J.Eggermont,E.G.P.Bovenkamp,J.Dijkstra, J.H.C.Reiber}@lumc.nl

Abstract. This paper discusses a study towards dynamic fitness based partitioning in IntraVascular UltraSound (IVUS) image analysis. Mixed-Integer Evolution Strategies (MI-ES) have recently been successfully used to optimize control parameters of a multi-agent image interpretation system for IVUS images lumen detection. However, because of complex interpretation contexts, it is impossible to find one single solution which works well on each possible image of each possible patient. Therefore it would be wise to let MI-ES find a *set* of solutions based on an optimal partition of IVUS images. Here a methodology is presented which does dynamic fitness based partitioning of the data during the MI-ES parameter optimization procedure. As a first step we applied this method to a challenging artificial test case which demonstrates the feasibility of our approach.

1 Introduction

IntraVascular UltraSound (IVUS) is a technique used to get real-time high resolution tomographic images from the inside of coronary vessels and other arteries. To gain insight into the status of an arterial segment a so-called catheter pull-back sequence is carried out. An example of an IVUS image with several detected features can be seen in Figure 1.

IVUS images are difficult to interpret, which causes manual segmentation to be highly sensitive to intra- and inter-observer variability. In addition, manual segmentation of the large number of IVUS images per patient is very time consuming. Therefore an automatic multi-agent image interpretation system was developed. However, feature detectors in this system consist of a large number of parameters that are hard to optimize, and there are continuous as well as different types of discrete parameters involved. Moreover, these parameters are subject to change when something changes in the image acquisition process.

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 391–398, 2007.
© Springer-Verlag Berlin Heidelberg 2007

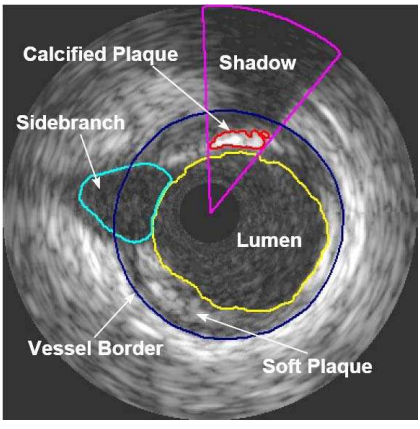


Fig. 1. An IntraVascular UltraSound (IVUS) image with detected features. The black circle in the middle is where the ultrasound imaging device (catheter) was located. The dark area surrounding the catheter is called the *lumen*, which is the part of the artery where the blood flows. Above the catheter *calcified plaque* is detected which blocks the ultrasound signal causing a dark *shadow*. Between the inside border of the vessel and the lumen there is some soft plaque, which does not block the ultrasound signal. The dark area left of the catheter is a sidebranch.

In [1,5] we compared the ability of Mixed-Integer Evolution Strategies (MI-ES) and standard Evolution Strategies (ES) to find optimal parameter settings for the detection of the lumen boundary in IntraVascular UltraSound (IVUS) images. Mixed-integer Evolution Strategies (MI-ES) are a special type of evolutionary strategy that can handle mixed-integer parameters (continuous, ordinal discrete, and nominal discrete) by using different mutation operators for the different types of parameters. Our results showed that the parameter solutions evolved by our MI-ES and ES algorithms were better than the default parameter solutions currently used, but also demonstrated that different sets of images require different parameter solutions for an optimal lumen segmentation.

Because of the complexity and variability of IVUS images, different parameters settings are needed for different image segmentation contexts. An ideal solution would be to classify IVUS images into different image segmentation contexts and optimize parameters for each context separately. Unfortunately, we do not know the number of IVUS image segmentation contexts as well as their characteristics. Additionally, unlike in the case of data points in a metric space, we have no natural distance measure [3] to cluster IVUS images into groups that need similar parameter settings for an optimal segmentation result. Only their degree of belonging to a partition, characterized by a particular set of parameters, can be measured by means of a training error for that image.

A possible approach for this kind of multi-level optimization problem could be cooperative coevolution (e.g., see [6,7]) in which we evolve both a set of parameter solutions and sets of images at the same time. However, this approach

requires a large number of fitness evaluations which is very computationally (and thus time) intensive and therefore not attractive to our problem.

Therefore we want to dynamically partition the IVUS image training set during the MI-ES parameter optimization process into groups of images which require a similar parameter settings for an optimal lumen segmentation result. Each group of images would correspond to a similar image segmentation context (for the image segmentation algorithm) and have an optimal parameter solution.

Aiming for the solution to the aforementioned problems we propose a multi-level optimization technique. Given a set of parameter solutions, we can partition the images according to which solution gives the best segmentation result. The fitness measure is then used as a "distance metric" to determine which partition (and corresponding MI-ES solution) is the best match for an IVUS image. By alternating partitioning and parameter optimization for each partition, images are dynamically repartitioned and parameter solutions optimized.

Our paper is structured as follows: Fitness based partitioning will be introduced in Section 2. This approach will then be applied to an artificial test problem, and first experimental results will be presented in Section 3. Conclusions and future work are given in Section 4.

2 Dynamic Fitness Based Partitioning

In general the multi-level optimization task is to find a proper fit of partitioning comprising N_P partitions, and for each of the partitions P_k ($k \in [1, N_P]$) we search for parameter settings which will result in an optimal solution for all problem instances in P_k. More concretely, in the case of IVUS lumen segmentation we try to find a partitioning of all IVUS images, and for each image partition we look for a parameter solution which results in the best possible lumen segmentation for those images. In order to solve this multi-level optimization problem we designed a 2-level algorithm with an inner and an outer loop.

In the outer loop the goal is to redistribute problem instances in order to achieve an improved global quality and to balance the size of the partitions. Aiming for this, a deterministic approach will be employed to determine how problem instances are (re-)partitioned and when to split or merge partitions.

In the inner loop the aim is to optimize parameter solutions for the problem instances in each of the N_P partitions. This task will be performed by evolutionary algorithms, in our case Mixed-Integer Evolution Strategies are used, since they can handle different parameter types simultaneously.

2.1 Algorithm

The detailed procedure for this 2-level optimization method is described by Algorithm 1 in the remainder of this section.

During the initialization phase all the problem instances (e.g., lumen images) are distributed over the N_P partitions, either randomly or based on some heuristics. Next a MI-ES algorithm MI-ES$_k$ is assigned for each partition P_k.

Algorithm 1. Fitness based partitioning algorithm

```
Divide problem instances I randomly over the partitions. /* Initialization */
while partitioning of problem instances keeps changing do
  for each partition P_k do
    run MI-ES_k on P_k for T iterations.
  end for
  for each MI-ES_k do
    select best individual/solution s_k
    apply best individual/solution s_k to all problem instances in I
  end for
  for each problem instance i do
    redistribute i to the partition P_k for which s_k offered the best solution.
  end for
  /* Split & Merge Partitions according to some heuristics */
  if the smallest partition P_S is empty then
    while the smallest partition P_S is empty do
      copy the population of MI-ES_L of the largest partition P_L to MI-ES_S
      divide the problem instances of P_L over P_L and P_S.
    end while
  else
    if the smallest partition P_S < 25% of the largest partition then
      divide problem instances of the smallest partition over other partitions
      copy the population of MI-ES_L of the largest partition P_L to MI-ES_S
      divide the problem instances of P_L over P_L and P_S.
    end if
  end if
end while
```

After the initialization phase the outer loop starts by applying each MI-ES algorithm to its corresponding partition. After all MI-ES algorithms have run for a fixed number of iterations the best solution s_k found by each MI-ES algorithm is applied to all the other problem instances. Based on which of the "best solutions" is the best for a particular problem instance i, each problem instance i is then redistributed to the partition whose MI-ES algorithm offers the best solution.

After all the problem instances have been re-assigned to the different partitions, next, heuristics are employed to handle empty and small partitions. In the current implementation we use a rather simple approach as described below, but in the future we want to extend this to a system which, through some kind of self-organization, can determine when to merge or split partitions in order to reach an optimal number of partitions.

Empty partitions are not useful, since their MI-ES algorithms cannot optimize anything. Therefore we replace the populations of MI-ES algorithms associated with empty partitions with a copy of the populations of the MI-ES algorithm of the largest partition. Additionally, half the problem instances of the largest

partition are put in the empty partition. This effectively both removes a non-useful empty partition and splits a large partition into two.

Small partitions can lead to overfitting of the parameter vectors. Small partitions are therefore treated similar to empty partitions. Their problem instances are redistributed over the other partitions (based on which partition is closest), and the resulting empty partition is treated as described above. Heuristically, we define any partition smaller than 25% of the size of the largest partition, as a small partition.

3 Artificial Test Problems and Results

In this paper we test the feasibility of "fitness based partitioning" on artificial problems as a first step toward application in the real IVUS feature detector system. This is done because testing out various algorithm settings and learning about its behavior using IVUS images is computationally too demanding to be practical. However, our test problems are designed in such a way that success may be expected on real problems, for instance, the data used in the test problems are representative for real cases.

The basic idea of our test setup, as visualized in Figure 2, is the task of finding a set of multidimensional distributions based on given data points. Two parts of the test problem need to be distinguished: (1) initialization/setup phase, (2) evaluation of a solution. Next, we give a brief description of both phases, followed by the detailed description of experiments and results.

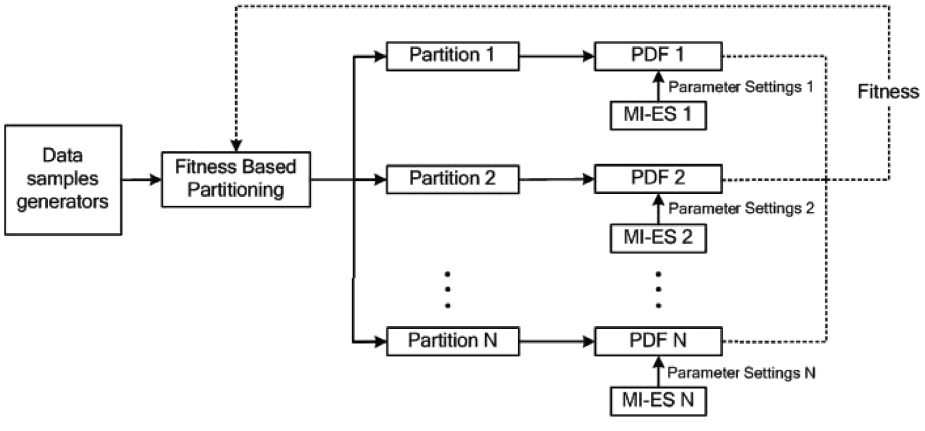


Fig. 2. Fitness based partitioning for randomly generated data samples

3.1 Initialization/Setup

In the *initialization/setup phase*, the problem generator creates sample points in D-dimensional space using a random number generator which can generate

values using either a uniform or normal distribution. Using the problem generator we created N_P "clusters" of sample points.

In more detail, the initialization procedure samples a set of N_I points $\mathcal{I} = \{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N_I)}\} \in (\mathbb{R}^D)^{N_I}$. The points are realizations of N_P different D - dimensional random variables $X_1, \ldots, X_{N_P}$. For each random variable N_I/N_P points are generated independently. For any $k \in [1, N_P]$, the distribution of the random variable X_k is determined by the parameters $\mu_d^{(k)}$, $\sigma_d^{(k)}$. The distribution of each random variable is an independent joint distribution composed of uniform and normal distributions. The values at the odd vector positions are sampled from 1-D normal distributions with mean value $\mu_d^{(k)}$ and standard deviation $\sigma_d^{(k)}$. The values at the even vector positions are sampled from 1-D uniform distributions with interval width $4\sigma_d^{(k)}$ and mean value $\mu_d^{(k)}$.

3.2 Evaluation

The test problem is to estimate the parameters and distribution types of the N_P multivariate distributions based on the initialized data points. We work with the following representation of solutions, encoded in the individuals of the EA. For each dimension $d \in [1, D]$ an individual has three parameters: an estimated mean value $\hat{\mu}_d \in \mathbb{R}$, an estimated standard deviation or, in case of uniform distribution, interval width $\hat{\sigma}_d \in \mathbb{R}$ and an estimated distribution type $\hat{\tau}_d$ (0: uniform, 1: normal). In case of a uniform distribution the minimum and maximum possible value are now defined as $\hat{\mu}_d^{(k)} - 2\hat{\sigma}_d^{(k)}$ and $\hat{\mu}_d^{(k)} + 2\hat{\sigma}_d^{(k)}$ respectively. Thus each individual looks like: $(\hat{\mu}_1^{(k)}, \hat{\sigma}_1^{(k)}, \hat{\tau}_1^{(k)}, \ldots, \hat{\mu}_d^{(k)}, \hat{\sigma}_d^{(k)}, \hat{\tau}_d^{(k)}, \ldots, \hat{\mu}_D^{(k)}, \hat{\sigma}_D^{(k)}, \hat{\tau}_D^{(k)})$. where k denotes the partition the individual represents.

For the fitness function we use a maximum log-likelihood approach, whereby for each individual the fitness is calculated as:

$$\text{fitness} = \sum_{i=1}^{|P_k|} \sum_{d=1}^{D} \log[PDF_{\hat{\mu}_d^{(k)}, \hat{\sigma}_d^{(k)}, \hat{\tau}_d^{(k)}}(\mathbf{x}_d^{(\underline{k}_i)})], \tag{1}$$

where $\mathbf{x}_d^{(\underline{k}_i)}$ denotes the d-th dimensional value of the i-th sample point from partition P_k.

The probability density function $PDF_{\hat{\mu}_d^{(k)}, \hat{\sigma}_d^{(k)}, \hat{\tau}_d^{(k)}} : \mathbb{R} \to [0, 1]$ is described as:

$$PDF_{\hat{\tau}_d^{(k)}} = \begin{cases} \mathrm{I}(\mathbf{x}_d^{(\underline{k}_i)} \in [\hat{\mu}_d^{(k)} - 2\hat{\sigma}_d^{(k)}, \hat{\mu}_d^{(k)} + 2\hat{\sigma}_d^{(k)}]) \frac{\mathbf{x}_d^{(\underline{k}_i)} - \hat{\mu}_d^{(k)}}{4\hat{\sigma}_d^{(k)}} & \hat{\tau}_d^{(k)} = 0 (\text{uniform}) \\ \frac{1}{2\pi} \exp(-\frac{\mathbf{x}_d^{(\underline{k}_i)} - \hat{\mu}_d^{(k)^2}}{2(\hat{\sigma}_d^{(k)})^2}) & \hat{\tau}_d^{(k)} = 1 (\text{normal}) \end{cases}$$

with $d = 1, \ldots, D$, $i = 1, \ldots, |P_k|$, $k = 1, \ldots, N_P$, and $\mathrm{I} : \{true, false\} \to \{0, 1\}$ being the indicator function: $\mathrm{I}(true) = 1, \mathrm{I}(false) = 0$.

3.3 Experimental Results

The MI-ES algorithms were programmed using the *Evolving Objects* library (EOlib) [4]. EOlib is an Open Source C++ library for all forms of evolutionary

computation and is available from http://eodev.sourceforge.net. The test-data generator was created using the random number generator from EOlib. For each combination of dimensionality and number of clusters we created 10 problem instantiations and on each problem instantiation we ran the fitness-based partitioning system 20 times using different random seeds for the MI-ES algorithms. Each generated cluster consists of 100 sample points.

For the MI-ES algorithms we used a plus-strategy with a population size of 40 and an offspring size of 280. After each redistribution cycle MI-ES algorithms were run for T iterations, with T dependent on the dimension D of the sample points. T was set to 50 for $D = 2$, to 75 for $D = 4$, and to 100 for $D = 6$.

Table 1. The results of the different experiments. Iterations outer loop (successful runs) means that all the N-dimensional data points that were originally created in a cluster end up in the same partition. Since the MI-ES algorithms have to find the optimal distribution parameters for each dimension, the number of variables to optimize is three times the dimension of the data points to be partitioned. For the successful runs we have measured the average, minimum and maximum number of iterations as well as standard deviation (S.D.), until a stable partitioning was reached.

Dimensions	T	Partitions	Succesful/Total	Iterations Outer Loop			
D		N_P	runs	Average	S.D.	Minimum	Maximum
2	50	3	200/200	7.375	3.57	1	18
4	75	10	200/200	16.07	3.44	9	31
4	100	20	198/200	23.21	3.44	16	36
6	100	10	199/200	12.35	4.45	5	43
6	100	20	197/200	14.87	3.66	9	34

The results in Table 1 show that, in most cases the fitness-based partitioning system manages to evolve a combination of uniform and normal distributions to describe each cluster. However, for $D = 4$ and $N_P = 20$, the system fails in two cases. In the first case a partition of 101 and a partition of 99 sample points result (vs. 100 each). In the second case, one partition is split into 2 smaller partitions containing 50 sample points each, while 2 other partitions are merged into one larger partition with 200 sample points. For the 6 dimensional problem with 10 clusters the only failure was a single sample point that was mispartitioned as well.

4 Conclusions and Outlook

In this paper, we proposed a two-stage approach for fitness based partitioning, a task similar but not identical to distance based clustering.

This approach has been developed to tackle real world (IVUS lumen detector) parameter optimization problems. To test our approach we generated multi-dimensional problems, which are similar to our real world problems but are less computationally demanding. On these problems the algorithm performed well.

Therefore we conclude that it is a promising approach for our real world problems on which we will apply the presented algorithms in the near future.

Up to now, the repartioning is done by a rather simple heuristic. Though this heuristic seems to work well in the discussed test case, a topic for future research will be to find more sophisticated ways for this. Moreover, we are aiming for systems that automatically can determine the number of clusters. For this recently proposed multi-objective optimization techniques [2] look promising.

Acknowledgments

This research is supported by the Netherlands Organization for Scientific Research (NWO) and the Technology Foundation STW.

References

1. E.G.P. Bovenkamp, J. Eggermont, R. Li, M.T.M. Emmerich, Th. Bäck, J. Dijkstra, and J.H.C. Reiber. Optimizing IVUS Lumen Segmentations using Evolutionary Algorithms. In *Medical Image Computing and Computer-Assisted Intervention*, Kopenhagen, Denmark, 2006.
2. J. Handl and J. Knowles. An investigation of representations and operators for evolutionary data clustering with a variable number of clusters. In *Parallel Problem Solving from Nature - PPSN IX, 9th International Conference*, volume 4193, pages 839–849, 2006.
3. A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: a review. *ACM Comput. Surv.*, 31(3):264–323, 1999.
4. M. Keijzer, J. J. Merelo, G. Romero, and M. Schoenauer. Evolving objects: a general purpose evolutionary computation library. In *EA-01, Evolution Artificielle, 5th International Conference in Evolutionary Algorithms*, 2001.
5. R. Li, M.T.M. Emmerich, J. Eggermont, E.G.P. Bovenkamp, Th. Bäck, J. Dijkstra, and J.H.C. Reiber. Mixed-integer optimization of coronary vessel image analysis using evolution strategies. In *Genetic and Evolutionary Computation Conference, GECCO, Proceedings*, pages 1645–1652, 2006.
6. L.Vanneschi, G. Mauri, A. Valsecchi, and S. Cagnoni. Heterogeneous cooperative coevolution: strategies of integration between gp and ga. In *Genetic and Evolutionary Computation Conference, GECCO, Proceedings*, pages 361–368, 2006.
7. M.E. Roberts and E. Claridge. Cooperative coevolution of image feature construction and object detection. In *Parallel Problem Solving from Nature - PPSN VIII, 8th International Conference*, volume 3242, pages 902–911, 2004.

Comparison Between Genetic Algorithms and the Baum-Welch Algorithm in Learning HMMs for Human Activity Classification*

Óscar Pérez[1,2], Massimo Piccardi[1], Jesús García[2],
Miguel A. Patricio[2], and José M. Molina[2]

[1] Faculty of Information Technology
University of Technology, Sydney
{oscapc,massimo}@it.uts.edu.au
[2] Computer Science Department-GIAA
Universidad Carlos III de Madrid
Colmenarejo, Spain
{opconcha,jgherrer,mpatrici}@inf.uc3m.es, molina@ia.uc3m.es

Abstract. A Hidden Markov Model (HMM) is used as an efficient and robust technique for human activities classification. The HMM evaluates a set of video recordings to classify each scene as a function of the future, actual and previous scenes. The probabilities of transition between states of the HMM and the observation model should be adjusted in order to obtain a correct classification. In this work, these matrixes are estimated using the well known Baum-Welch algorithm that is based on the definition of the real observations as a mixture of two Gaussians for each state. The application of the GA follows the same principle but the optimization is carried out considering the classification. In this case, GA optimizes the Gaussian parameters considering as a fitness function the results of the classification application. Results show the improvement of GA techniques for human activities recognition.

1 Introduction

The problem addressed in this paper is the recognition of human activities by analysis of features extracted from a video tracking process, with a special emphasis on data modelling and classifier performance. Categorization of human activity is an active research area in computer vision and covers several areas such as detection, motion analysis, pattern recognition, etc. The usual tasks involved are people detection, identification, motion-based tracking, and activity classification. The final goal is to cover the gap between the low-level tracking information provided by the system and the interpretation necessary to have an abstract description of the scene. This capability is needed as a practical means to avoid the human burden on operator continuously monitoring video cameras.

* Funded by projects CICYT TSI2005-07344, CICYT TEC2005-07-186 and CAM MADRINET S-0505/TIC/0255.

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 399–406, 2007.
© Springer-Verlag Berlin Heidelberg 2007

Typically, in order to recognize human behaviours, the detection of moving objects is not enough, but an analysis of trajectory and interaction with the scene context (other objects and elements) is necessary to infer the behaviour type [1-2]. Therefore, an important requirement of these systems is the capability to track and maintain identity of all detected objects over time.

The process basically involves matching a measured sequence of observations to a certain set of states (or sequences of states) representing predefined actions. There are several approaches for matching time-varying data. Dynamic time warping (DTW) [3, 4], HMM (hidden Markov models) [5-7], Bayesian networks [8] and declarative models [9]. Human actions are represented by sequences of states which must be inferred from features extracted from images (shapes, body parts, poses, etc.) and attributes extracted from motion such as velocity, trajectory, etc. In this work, three states are used corresponding to INACTIVE, WALKING and RUNNING activities. The features selected to classify these actions are the velocities measured in the scene.

The first step of the application is the learning task. The main problem associated with HMMs is to take a sequence of observations (the speed measurements mentioned above), known to represent a set of hidden states, and fit the most probable HMM [10]; that is, determine the parameters that most probably describe what occurs in the scene. The forward-backward algorithm (Baum-Welch, which is an expectation maximization (EM) algorithm) is used when the HMM parameters are not directly (empirically) measurable, as is very often the case in real applications. The second step is the decoding task, finding the most probable sequence of hidden states given some observations that is, finding the hidden states that generated the observed output. The sequence of states obtained by the Viterbi algorithm is then compared with the ground truth to measure the accuracy.

In this work, we propose a Genetic Algorithm to adjust the HMM matrixes. The fitness function is the classification performance of the second step. We later compare its accuracy against that obtained by the Baum-Welch algorithm for the same task. We prove that GA outperforms the traditional Baum-Welch in the estimation of the parameters, resulting in higher accuracy in the state decoding. In our experiments, the GA achieved 91.11% accuracy while the EM 74.32 %.

In the next section, the HMM structure and Viterbi algorithm are outlined as selected framework to infer the most probable sequence of states according to observations. Section 3 focuses on free parameters of the model and alternative learning techniques with the EM and GA approaches. The fourth section adjusts the parameters of the HMM by means of the two algorithms, and the results are discussed in the fifth section.

2 Hidden Markov Model

The Hidden Markov Model (HMM) [10] is a model framework with state transitions and observation distributions used to infer the sequence of states from the available features extracted from images. It is defined by the triple $\lambda = (A, B, \Pi)$. A are called the state transition probabilities and express the Markovian hypothesis that the

value, i, of the current state, S_k, is only dependent on the value, j, of the previous state, S_{k-1}. B are called the observation probabilities, quantifying the probability of observing the O_k value when the current state is j. Eventually, π are called the initial state probabilities and quantify the probabilities of values for the initial state. The actual history of the human's activity states versus time is retrieved using the Viterbi algorithm. They definition and process are characterized by the following elements:

- The target locations as well as the sensor observations are modelled by a set of states, $S = \{S_1, S_2, \cdots, S_N\}$.
- The set of time-invariant transition probabilities

$$a_{ij} = P[q_k = S_j \mid q_{k-1} = S_i], \quad 1 \le i, j \le N \tag{1}$$

where q_k denotes the actual state at time k ($1 \le k \le T$)

- The observation probability distribution

$$b_i(O_k) = P[O_k \mid q_k = S_i], \quad 1 \le i \le N \tag{2}$$

where O_k denotes the observation state at time k.

- The initial state distribution

$$\pi_i = P[q_1 = S_i], \qquad 1 \le i \le N \tag{3}$$

The Viterbi algorithm is then used to find the best state sequence, $q_1^*, q_2^*, \cdots, q_T^*$, given the observation sequence $O_1, O_2, \cdots, O_T$. The highest probability along a single path, which accounts for the first k observations and ends in S_i at time k, is defined as:

$$\delta_k(i) = \max_{q_1, q_2 \cdots, q_{k-1}} P[q_1 q_2 \cdots q_k = S_i, O_1 O_2 \cdots O_k \mid \lambda]. \tag{4}$$

It can be induced that

$$\delta_{k+1}(j) = \max_i [a_{ij} \delta_k(i)] \cdot b_j(O_{k+1}) \tag{5}$$

The best state sequence can be retrieved by keeping track of the argument that maximizes previous equation for each k and j. The observation probability distribution, $b_j(O_k)$, is a Gaussian mixture likelihood function as mentioned above.

3 Comparison Between the Baum-Welch Algorithm and GA

The well-known Baum-Welch algorithm can be used to simultaneously estimate the state transition probabilities and the observation probabilities in a maximum likelihood framework from the sequence of observations. The states in HMM are assume as discrete values. However, the observations can have either discrete (limited) values, modelled with probabilities in matrix B, or continuous (or continuous discretised), modelled by conditional probability density functions. In the latter case,

HMMs can use MoGs to model their observation probabilities. Each state's value, i = 1..N, has its own MoG, independent from those of the other values. Thus, for example, the observation likelihood for i-th state, $b_i(O_k)$, is given by the following MoG:

$$b_i(O_k) = \sum_{l=1}^{M} \alpha_{il} G(O_k, \mu_{il}, \sigma_{il}^2) \tag{6}$$

Then, we can define $p(l \mid O_k, \Theta)$ as the probability of the l-th Gaussian component at sample O_k:

$$p(l \mid O_k, \Theta) = \frac{\alpha_{il} G(O_k, \mu_{il}, \sigma_{il}^2)}{\sum_{p=1}^{M} \alpha_{ip} G(O_k, \mu_{ip}, \sigma_{ip}^2)} . \tag{7}$$

The weights, means and variances ($\alpha_{il}, \mu_{il}, \sigma_{il}^2$) of the observation distributions (the B "matrix"), are then calculated over the set of observed values, O_k, k = 1..T. The numerators and denominators are modulated (i.e. multiplied) by the probability of being in state i at time k, $\gamma_i(k)$

$$\alpha_{il}^{new} = \frac{\sum_{t=1}^{T} p(l \mid O_k, \Theta)\, \gamma_i(k)}{\sum_{k=1}^{T} \gamma_i(k)} \qquad \mu_{il}^{new} = \frac{\sum_{k=1}^{T} O_k \; p(l \mid O_k, \Theta)\, \gamma_i(k)}{\sum_{k=1}^{T} p(l \mid O_k, \Theta)\, \gamma_i(k)} \tag{8-9}$$

$$\sigma_{il}^{2\,new} = \frac{\sum_{k=1}^{T} (O_k - \mu_{lt})^2 \; p(l \mid O_k, \Theta)\, \gamma_i(k)}{\sum_{k=1}^{T} p(l \mid O_k, \Theta)\, \gamma_i(k)} \tag{10}$$

Finally, the problem could be stated as: How to calculate these three parameters? In the first case, using an EM algorithm [11], we calculate μ, σ and α, but some problems appear:

1. This algorithm is very sensitive to the initialisation of the parameters
2. We always get a local optimum.
3. The algorithm does not use the ground truth data.

In the second case, we use a GA to calculate the μ, σ and α parameters. This algorithm is not so very sensitive to the initialisation of the parameters, because GA begins with many individuals and mutation operator generates new individuals in different search zones. This characteristic allows to the GA to avoid getting the local optimum and hopefully finding a global optimum. Moreover, during the training process we know exactly how to match the output of the algorithm with the ground truth and use this as the fitness function.

4 Methodology and Results

The aim of our experiments was the classification of human activities by means of a HMM, carrying out a comparison between the Baum-Welch and Genetic Algorithm used for the adjustment of the parameters of the HMM. Moreover, we desired to use the minimum number of features for the easy and fast performance of the HMM.

We used the CAVIAR video dataset [12] and selected two long videos in which several actors behaved in different ways: "Two people meet, fight and run away" and "Two people meet, fight, one down, other runs away". Among all the activities showed in these videos we reduced the set to three: Inactive ("in"), Walking ("wk") and Running ("r").

The ground truth was found by hand-labelling and we must take into account for the final results the subjectivity when classifying, especially between the classes walking and running. Therefore, it is important to highlight that the ground truth is made up of sequences that show the activity of each actor from the point of their appearance until they disappear.

Finally, the whole programming was developed in Matlab.

The features selected to classify the actions are the velocities measured from real cases. The velocities are calculated considering the position of the human in different frames. In particular, we computed the speed at 5 and 25 frame intervals as follows:

$$speed_f = \frac{1}{f}\sqrt{(x_i - x_{i-f})^2 + (y_i - y_{i-f})^2} \tag{11}$$

where f is set to 5 and 25, respectively, and (x_i, y_i) and (x_{i-f}, y_{i-f}) are the subject's positions in the image plane.

Thus, the first experiments were the adjustment of the HMM parameters using the Baum-Welch algorithm and the Kevin Murphy Matlab toolbox [13]. The prior probability is fixed to be [state 1=1; state 2= 0; state 3 = 0], which means that we assume that the first state is always inactive. This assignment results very useful when decoding the sequence with Viterbi, as we do not know a priori the correspondence between the states in the Viterbi output and those in the ground truth. By fixing this probability we assure that the first code of the Viterbi output will match the inactive state.

The initialization of the variables for the EM algorithm influences greatly in the final result. Hence, although the first option was to initiate randomly the different variables, the poor results obtained made us initialise the variables by taking into account the primitive knowledge of the observation data. Therefore, for our experiments (three states and three Gaussians per state) the initialisation of the 51 variables was fixed like as such:

- Means (12 parameters) and covariances (24 parameteres): μ_i = [0.1; 2.0; 5.0] $\sigma_i = 1$
- Weight for each of the Gaussians (6 parameters): random

- State transition matrix (9 parameters): a_{ij} =[0.8 0.19 0.01; 0.1 0.8 0.1; 0.01 0.8 0.19] where the transition from inactive to running (and vice versa) is very small.
- Maximum number of iterations for the EM algorithm: *max-iter=50*

The result of the EM algorithm gave the State transition matrix and the Observation probability matrix that define the HMM. Then, we use the Viterbi algorithm to determine the most probable sequence of hidden states given a sequence of observations as input of our HMM and measure accuracy against the ground truth.

The second set of experiments adjust the parameters of the HMM by means of a Genetic Algorithm and the Matlab Genetic Algorithm Toolbox of the University of Sheffield [14]. The individual is made up of the 51 parameters mentioned above.

Again, the prior probability is fixed to [state 1=1; state 2= 0; state 3 = 0] in order to have the same conditions for both experiments.

Nevertheless, the initiation of the individuals is carried out randomly since the GA is not as sensitive to the initiation as the EM algorithm was. Other important parameters of the GA are fixed as follows:

- No. of individuals: NIND = 45;
- Generation gap: GGAP = 0.8 which means that NIND x GGAP=36 are produced at each generation.
- Crossover rate and mutation rate: XOVR = 1; MUTR = 1/(41);
- Maximum no. of generations: MAXGEN = 500;
- Insertion rate: INSR = 0.9 that specifies that 90% of the individuals produced at each generation are reinserted into the population.

Each individual of the population is validated with the Viterbi algorithm. As a result, the fitness function is defined as the comparison between the output of the Viterbi algorithm and the ground truth.

The data are divided into two big groups of sequences (Holdout cross-validation): one of 8 sequences of 2478 data in total and another one of 5 sequences and 1102 data. The first one is used to train while the second validates the learned HMM. The table 1 shows the total error for the training and validation for the EM and the GA.

Table 1. Total Error of classification for the training and validation data

Total Error (%)	**Training**	**Validation**
EM	20.82	25.68
GA	10.17	8.89

Table 2. Confusion matrix for the classification with the EM and GA algorithms for the validation data

EM	Predicted			
Actual		in	wk	r
	in	550	1	0
	wk	12	179	269
	r	0	1	90

GA	Predicted			
Actual		in	wk	r
	in	551	0	0
	wk	6	453	1
	r	0	91	0

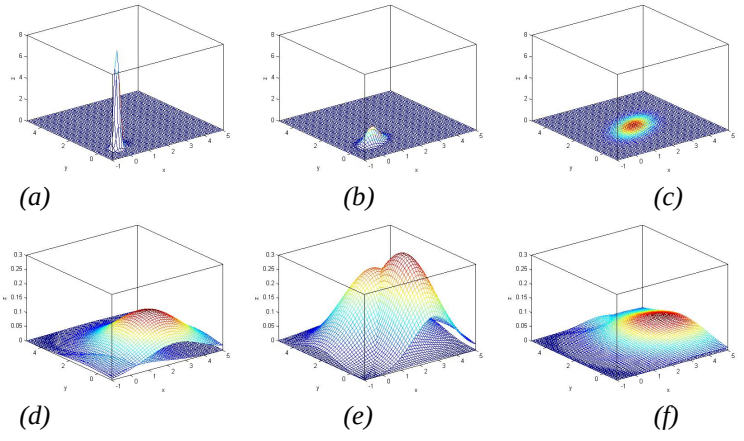


Fig. 1. Mixture of Gaussians derived from the EM algorithm (in a scale 0-8) for inactive (a), walking (b) and running (c). Mixture of Gaussians derived from the GA algorithm (in a scale 0-0.3) for inactive (d), walking (e) and running (f).

The next table show the confusion matrix for the EM and the GA where each column of the matrix represents the instances in a predicted class, while each row represents the instances in an actual class.

5 Discussion

The results show a clear better performance for the GA over the EM algorithm. Furthermore, the EM algorithm does not take advantage of having two Gaussian distributions per state and merges both functions in only one.

The EM inferred a Gaussian that classifies 90 out 91 running samples in a correct way, but paying the high price of misclassifying 269 walking samples as running. Nevertheless, the genetic algorithm enhances the total classification by ignoring the running samples and classified in a cored way 453 walking samples out of 454.

As we can see in the pictures of the Gaussians, the EM inferred a great peak for the inactive activity, in contrast with the low probability assigned to the running state. On the other hand, the GA creates a small probability for the inactivity. In fact, the first mode of the Gaussian for the inactive state is centred in (0.5322, 0.2755) with a small weight of 0.5876. The second mode is centred in (2.3928, 0.8261) and a weight of 2.11. Nevertheless, the GA creates a high probability distribution for the walking state with two Gaussian centred in (2.0002, 2.2480) and (2.7381, 1.0947) and weight of 3 and 4 respectively.

Moreover, the result of the classification by using EM is greatly dependant on the initialisation of the parameters in contrast to the GA that give stable results independently of the initial values.

As a conclusion, we can derive that for this case the GA outperforms the traditional EM at the expense of ignoring the running activity. Thus, for real applications, a trade-off must be done according to the results that we are interested in obtaining.

References

1. Moeslund, T.B. and Granum, E. 'A Survey of Computer Vision-Based Human Motion Capture'. Computer Vision and Image Understanding, Volume 81(3), 2001, pp. 231 – 268.
2. Pavlidis, I., Morellas, V., Tsiamyrtzis, P., and Harp, S. 'Urban surveillance systems: from the laboratory to the commercial world', Proc. IEEE, 2001, 89, (10), pp. 1478–1495.
3. Rath, T.M., and Manmatha, R.: 'Features for word spotting in historical manuscripts'. Proc. of the 7th Int. Conf. on Document Analysis and Recognition, 2003, pp. 512–527.
4. Oates, T., Schmill, M.D., and Cohen, P.R.: 'A method for clustering the experiences of a mobile robot with human judgements'. Proc. of the 17th National Conf. on Artificial Intelligence and Twelfth Conf. on Innovative Applications of Artificial Intelligence, AAAI Press, 2000, pp. 846–851.
5. Yamato, J., Ohya, J., and Ishii, K., 'Recognizing human action in time-sequential images using hidden Markov model,' Proceedings of CVPR '92, pp. 379-385.
6. Wilson, A.D. and Bobick, A.F., 'Parametric hidden Markov models for gesture recognition', IEEE Transactions on Pattern Analysis and Machine Intelligence, 1999, 21, (9), pp. 884 – 900.
7. Bicego, M. and Murino, V., 'Investigating hidden Markov models' capabilities in 2D shape classification,' IEEE Transactions on Pattern Analysis and Machine Intelligence, 2004, 26, (2), pp. 281 – 286.
8. Nguyen, N.T., Bui, H.H., Venkatesh, S., and West, G.: 'Recognising and monitoring high-level behaviour in complex spatial environments'. IEEE Int. Conf. on Computer Vision and Pattern Recognition, Wisconsin, 2003, pp. 1–6.
9. Rota, N. and Thonnat, M. 'Activity Recognition from Video Sequences using Declarative Models', Proc. European Conf. on A.I., 2002, pp. 673 – 680.
10. Stamp M 'A Revealing Introduction to Hidden Markov Models' January 18, 2004, http://www.cs.sjsu.edu/faculty/stamp/RUA/HMM.pdf.
11. Bilmes, J. 1998. A gentle tutorial on the EM algorithm and its application to parameter estimation for Gaussian mixture and Hidden Markov Models. Tech. Rep. ICSI-TR-97-021, University of California Berkeley.
12. http://homepages.inf.ed.ac.uk/rbf/CAVIAR/caviar.htm
13. http://www.cs.ubc.ca/~murphyk/Software/HMM/hmm.html
14. http://www.shef.ac.uk/acse/research/ecrg/gat.html

Unsupervised Evolutionary Segmentation Algorithm Based on Texture Analysis

Cynthia B. Pérez and Gustavo Olague

CICESE, Research Center, División de Física Aplicada
Centro de Investigación Científica y de Educación Superior de Ensenada,
B.C., Km. 107 Carretera Tijuana-Ensenada, 22860, Ensenada, B.C., México
cbperez@cicese.mx,olague@cicese.mx

Abstract. This work describes an evolutionary approach to texture segmentation, a long-standing and important problem in computer vision. The difficulty of the problem can be related to the fact that real world textures are complex to model and analyze. In this way, segmenting texture images is hard to achieve due to irregular regions found in textures. We present our *EvoSeg* algorithm, which uses knowledge derived from texture analysis to identify how many homogeneous regions exist in the scene without *a priori* information. *EvoSeg* uses texture features derived from the Gray Level Cooccurrence Matrix and optimizes a fitness measure, based on the minimum variance criteria, using a hierarchical GA. We present qualitative results by applying *EvoSeg* on synthetic and real world images and compare it with the state-of-the-art JSEG algorithm.

1 Introduction

Human vision is a complex process that is not yet completely understood, despite several decades of studying the problem from the standpoint of natural science and artificial intelligence. In computer vision, the complex physical process of identifying colors, shapes, textures and automatically grouping them into separate objects within a scene continues to be an open research avenue. Image segmentation denotes a process by which an input image is partitioned into regions that are homogeneous according to some group of characteristics, i.e., texture information. Formally, image segmentation could be defined as follows: Segmentation of I is a partition P of I into a set of M regions R_m, $m = 1, 2, ..., M$, such that:

$$\begin{aligned} &1)\ \bigcup_{m=1}^{M} R_m = I \quad \text{with} \quad R_m \bigcap R_n = \emptyset, \quad m \neq n \\ &2)\ H(R_m) = true \quad \forall\ m \\ &3)\ H(R_m \bigcup R_n) = false \quad \forall\ R_m \ \text{and}\ R_n \ \text{adjacent} \end{aligned} \tag{1}$$

where I is the image and H is the predicate of homogeneity. Thus, each region in a segmented image needs to simultaneously satisfy the properties of homogeneity

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 407–414, 2007.
© Springer-Verlag Berlin Heidelberg 2007

and connectivity [1]. A region is homogeneous if all of its pixels satisfy an homogeneity predicate defined over one or more pixel attributes such as intensity, texture or color. On the other hand, a region is said to be connected if a connected path exists between any two pixels within the region.

Because of the large diversity of segmentations methods, it is difficult to review each individual image segmentation technique proposed up to now. Segmentation methods usually can be classified as region or boundary based [2], histogram-based [3] or graph-based [4], to mention but a few. In the evolutionary computer vision community there are also a number of works dealing with image segmentation [1,5,6,7,8]. The application of evolutionary techniques to image processing and computer vision have increased mainly due to the robustness of the approach [9]. In this paper we pose image segmentation as a combinatorial optimization problem. We present *EvoSeg* as an evolutionary image segmentation algorithm based on texture information extracted in a Gray Level Cooccurrence Matrix (GLCM). *EvoSeg* attempts to identify how many homogeneous regions exist in the scene without *a priori* information. We qualitatively compare our results with the JSEG algorithm [10].

The remainder of this paper is organized as follows. Section 2 describes the cooccurrence matrix and texture descriptors that are used for texture analysis. Section 3 introduces the *EvoSeg* algorithm giving emphasis to the explanation on how evolution was applied to the segmentation problem. Section 4 shows the results of the *EvoSeg* and JSEG algorithm, illustrating several comparative results. Finally, Section 5 gives some concluding remarks.

2 Texture Analysis

Texture analysis is a long-standing and important problem in computer vision. It comprises problems like texture classification, texture segmentation, texture synthesis and shape from texture. Obviously, a given application of texture analysis usually falls into more than one category; our work is focused on texture segmentation. The difficulty of the problem can be related to the fact that real world textures are complex to model and analyze. Historically, the most commonly used methods for describing texture information are statistical based approaches, which include first order, second order and higher order statistical methods. These methods analyze the distribution of a texture property for each pixel contained in the image. We are interested in second order statistical methods, which represent the joint probability density of the intensity values between two pixels separated by a given vector $\mathbf{V}$. This information is coded using the GLCM denoted by $M(i,j)$. The GLCM $M(i,j)$ describes the frequency of a gray value that appears in a specific spatial relationship with another gray value within a given window, denoted by $f(x,y)$.

Formally, the GLCM $M_{i,j}(\pi)$ defines a joint probability density function $f(i,j|\mathbf{V},\pi)$ where i and j are the gray levels of two pixels separated by a vector

$\mathbf{V}$, and $\pi = \{\mathbf{V}, R\}$ is the parameter set for $M_{i,j}(\pi)$. The GLCM identifies how often pixels that define a vector $\mathbf{V}\,(d, \theta)$, and it differs by a certain amount of intensity value $\Delta = i - j$ appearing in a region $\mathbf{R}$ of a given image $\mathbf{I}$.

The GLCM presents a problem when the number of different gray levels in region $\mathbf{R}$ increase, turning difficult to handle or use directly due to the dimensions of the GLCM. Fortunately, the information encoded in the GLCM can be expressed by a varied set of statistically relevant numerical descriptors. This reduces the dimensionality of the information that is extracted from the image using the GLCM. Extracting each descriptor from an image effectively maps the intensity values of each pixel to a new dimension. Descriptors extracted from $M(i, j)$ include the following: Entropy, Homogeneity, Local Homogeneity, Contrast, Moments, Inverse Moments, Uniformity, Maximum Probability, Correlation and Directivity [11,12]. Such descriptors may be defined in the spatial domain, like those extracted from the GLCM, or can be extracted in other frequency domains.

3 EvoSeg Algorithm

The evolutionary segmentation algorithm *EvoSeg* attempts to identify how many homogeneous regions exist in the scene without *a priori* information. The algorithm uses a fitness function that evaluates possible segmentations based on region homogeneity and region distinctiveness. The *EvoSeg* algorithm carries out two general process, a statistical texture analysis process and a segmentation process immerse into a genetic algorithm. The complete flow-chart of the algorithm is shown in Fig. 1(a).

3.1 Statistical Texture Analysis Process

The statistical texture analysis process is used as a way of obtaining representative compact data of the image texture through the GLCM and texture descriptors. In order to calculate the GLCM, we tested experimentally different windows sizes, directions and distances. The results showed only substantial differences when increasing the window size, it produces blurring descriptor images. We used the following parameters: the window size was set as 7×7 pixels, the direction as $0°$, and the distance as 1 pixel. The GLCM information is used to calculate the texture descriptors for each pixel of the image; in that way, we obtain one matrix for each texture descriptor. Different texture descriptors were used one by one, or as a combination of them. However, the descriptor which gave better experimental results in our study was the second order moment, which is defined as follows:

$$Mom_k = \sum_i \sum_j (i - j)^2 \cdot M(i, j) \tag{2}$$

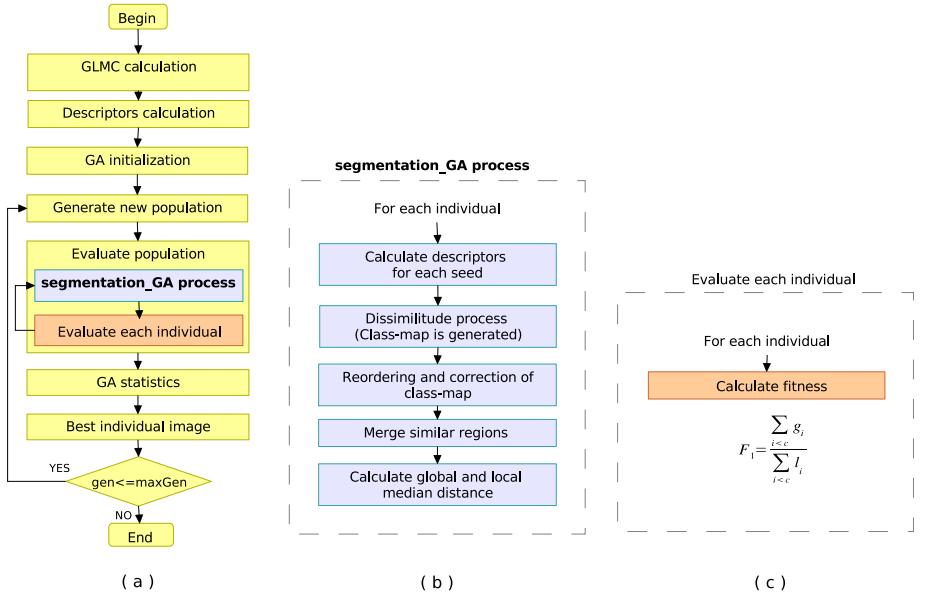


Fig. 1. *EvoSeg* Flow-chart. (a) Complete *EvoSeg* flow-chart. (b) General steps for segmentation process. (c) GA Fitness Function scheme.

3.2 Segmentation Process Immersed into a Genetic Algorithm

Our approach evaluates the image segmentation using a genetic algorithm to decide which segmented image is the best from the set of possible solutions based on the following algorithm, see Fig. 1(a):

1. **Initialize the genetic algorithm parameters.** The chromosome is coded with a hierarchical structure, it contains M elements representing the possible centroids of each image region denoted by C_i where $i = 1...M$ as shown in Fig. 2(a). The binary array, see Fig. 2(b), contains the control bits that indicates the final centroids to use in the segmentation process, see Fig. 2(c). We experimentally conclude that it is better to use the binary array because it allows a greater diversity of centroids in new individuals. Tournament was chosen as the selection method in the GA. The crossover is accomplished after the first generation of each individual with a probability of 90%. The mutation rate used was 10% with a population size of 50 and 30 generations.
2. **Segmentation process.** Once the algorithm has selected the possible regions, it is necessary to know how to classify the pixels. We use texture and spatial information in order to classify pixels of the possible regions. Suppose we have a set $P = \{x_1, x_2, ..., x_N\}$ of N pixels where N is the total amount of pixels. Let also have M disjoint subsets $R_1, R_2, ...R_M$ where each subset represents a region and has an associated centroid C_i. Moreover, we have a

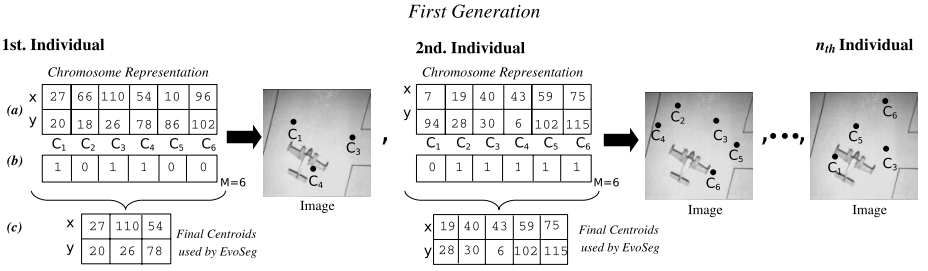


Fig. 2. Example of the chromosome representation in the first generation of the *EvoSeg* algorithm

set $D = \{d_1, d_2, ..., d_N\}$ of N descriptor values. In this way, the segmentation process is as follows, see Fig. 1(b):

(a) We calculate a descriptor value s_{C_i} for each initial centroid given by the mean of the descriptor values d_i within a 5×5 neighborhood around the centroid.

(b) We create an initial class map, that can be viewed as a special matrix indicating the region to which each pixel belongs. The class map is created using nearest neighbor classification. Two distances measures[1] are defined, Δ and δ. Δ is the distance in the descriptor space and δ is the distance within the image plane. In this way, a pixel x_j is assigned to a region R_i, if $\Delta(d_j, s_{C_i}) < \Delta(d_j, s_{C_l}) \ \ \forall\ l$ and if $\delta(x_j, C_i) < t$, where t is a threshold and $l = 1...M$ with $l \neq i$.

(c) The class map is rearranged and corrected with the purpose of improving the regions with a poor classification.

(d) Two regions R_i and R_j are merged if they satisfy a similarity criterion based on the median of each region. Then, during the segmentation process, before the fitness evaluation, the centroids are updated each time an element is added into a region, using the following expression:

$$centroid(x, y) = (\frac{\sum_{I \epsilon A} x_i}{\mid A \mid}, \frac{\sum_{I \epsilon A} y_i}{\mid A \mid}), \tag{3}$$

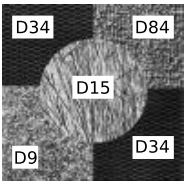
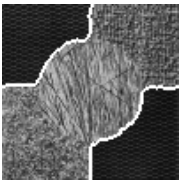
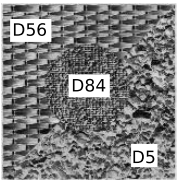
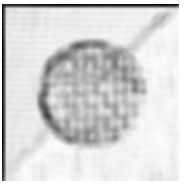
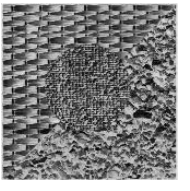
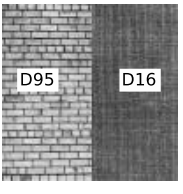
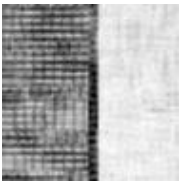
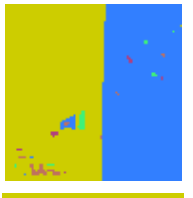
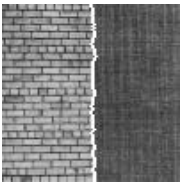
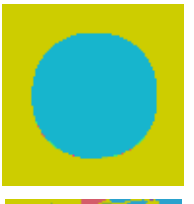
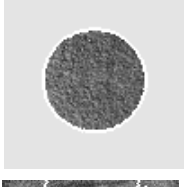
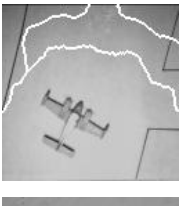
where x, y are the pixel coordinates in the image and $\mid A \mid$ is the number of elements in the region A. In this way, the genetic information of the chromosome is adjusted to facilitate the classification of image pixels.

3. **Fitness Evaluation.** The fitness function used by the genetic algorithm is based on local and global minimum distance measures between regions, see Fig. 1(c). The local distance is,

$$l = \sqrt{\sum_{i=1}^{c} \sum_{j=1}^{n_i} (d_j - m_i)^2}, \tag{4}$$

[1] All distances measures are Euclidean.

Table 1. *EvoSeg* and JSEG results using different texture images

	Original Image	Descriptor Image	*EvoSeg* Result	JSEG Result
(a)				
(b)				
(c)				
(d)				
(e)				
(f)				
(g)				

where d_j represents the descriptor values and m_i represents the median by region. The global distance is given by

$$g = \sqrt{\sum_{i=1}^{c} (m_i - m)^2}, \quad \text{where} \quad m = \frac{1}{N} \sum_{i=1}^{c} m_i \cdot n_i \tag{5}$$

m is the total median and n_i is the number of elements in the i^{th} region.

The proposed fitness function showed in Fig.1(c) indicates that the distances between the medians of different regions should be maximized in order to maintain the separability between regions. In addition, we have to minimize the distances between elements within a region because the elements nearest to a centroid should belong to the same region. This process is repeated while the maximum number of generations is reached.

4 Experimental Results

In this section, we present the experimental results obtained by EvoSeg and JSEG algorithms tested on seven images. The JSEG algorithm is a state-of-art segmentation algorithm that considers color and texture information in image and video, see [10]. The motivation to use this segmentation algorithm is due to its quality and simplicity, as well as its versatility on high level tasks.

Table 1 presents our experimental results, both algorithms produce good segmentations in images 1(a),(c),(d) and produce a misclassification of pixels in the image 1(g). The *EvoSeg* segmentation in Table 1(b) is better than JSEG, this is due to the fact that JSEG is sensitive to illumination changes. In this case, JSEG could not segment the image because the textures do not present high contrast. Otherwise, *EvoSeg* is robust to the lack of image contrast and almost achieves the complete segmentation of the texture D84. Table 1(e) shows the results for the baboon image, the segmentation done by the JSEG algorithm is superior due to the high contrast of the textures. On the other hand, *EvoSeg* presents problems segmenting the image because the contours of the descriptor image are not well defined. In this case, it would be interesting to add the JSEG color quantization method into the *EvoSeg* algorithm. The airplane shown in Table 1(f) is segmented by *EvoSeg* while JSEG was lost in the illumination. In general, the segmented images by *EvoSeg* are comparable with an algorithm that is state-of-art and it could be more robust by adding the color quantization method before generating the class map.

The images used in our experiments were obtained online at the USC-SIPI Image DataBase[2]. The original images in Table 1(a),(b),(c) and (d) are used by Yoshimura [7].

5 Conclusions

In this paper, we have presented *EvoSeg* as an unsupervised evolutionary segmentation algorithm. *EvoSeg* identifies a good image segmentation from multiple

[2] Signal and Image Processing Institute: http://sipi.usc.edu/database

solutions using a genetic algorithm. Our segmentation process depends on the second moment descriptor extracted from the GLCM. If the descriptor image is well defined, it is easier for the algorithm to identify the region boundaries. The hierarchical GA allowed a better distribution of the regions during the segmentation process, and produces a variety of good solutions from which we select the best one. In further research, it would be interesting to analyze the problem as multiobjective and to obtain a pareto of possible segmentations. We compared our results with the JSEG algorithm and both segmentations were comparable.

Acknowledgments. This research was funded by Ministerio de Ciencia y Tecnología, and by CONACyT and INRIA through the LAFMI project. First author supported by scholarship 0416442 from CONACyT.

References

1. Suchendra M.Bhandarkar and Hui Zhang. "Image Segmentation Using Evolutionary Computation". IEEE Transactions on Evolutionary Computation. Vol.3 (1). April 1999.
2. Timo Ojala and Matti Pietikäinen. "Unsupervised texture segmentation using feature distributions". Pattern Recognition. Vol.32. pp. 477-486. 1999.
3. Jan Puzicha, Thomas Hofmann and Joachim M. Buhmann. "Histogram Clustering for Unsupervised Segmentation and Image Retrieval". Pattern Recognition Letters. 1999.
4. Jianbo Shi and Jitendra Malik. "Normalized Cuts and Image Segmentation". IEEE Transactions on Pattern Analysis and Machine Intelligence. Vol.32 (8). 2000.
5. S. Cagnoni, A.B.Dobrzeniecki, R.Poli and J.C.Yanch."Genetic algorithm-based interactive segmentation of 3D medical images". Image and Vision Computing. Vol.17. pp.88-895. 1999.
6. Bir Bhanu, Sungkee Lee and John Ming. "Adaptive Image Segmentation Using a Genetic Algorithm". IEEE Transactions on Systems, Man, and Cybernetics. Vol.25 (12). December 1995.
7. Motohide Yoshimura and Shunichiro Oe. "Evolutionary segmentation of texture image using genetic algorithms towards automatic decision of optimum number of segmentation areas. Pattern Recognition. Vol.32. pp.2041-2054. 1999.
8. Leonardo Bocchi, Lucia Ballerini and Signe Hässler. "A new evolutionary algorithm for image segmentation". Applications on Evolutionary Computation. LNCS 3449. pp. 264-273. 2005.
9. Olague, G., Cagononi, S., and Lutton, E. (eds). "Introduction to the Special Issue on Evolutionary Computer Vision and Image Understanding". Pattern Recognition Letters. Vol.27 (11). pp.1161-1163, 2006.
10. Yining Deng and B.S.Manjunath. "Unsupervised Segmentation of Color-Texture Regions in Images and Video". IEEE Transactions on Pattern Analysis and Machine Intelligence. Vol.23 (8). pp. 800-810. August 2001.
11. J.R. Parker. "Algorithms for Image Processing and Computer Vision". John Wiley, New York. 1996.
12. R.M.Haralick, K. Shanmugam and I. Dinstein. "Textural features for image classification". IEEE Transactions on System, Man and Cibernetics. Vol.3 (6). pp.610-621. 1973.

Evolutionary Approaches for Automatic 3D Modeling of Skulls in Forensic Identification*

J. Santamaría[1], O. Cordón[2,3], and S. Damas[2,4]

[1] Dept. Software Engineering, University of Cádiz, Cádiz, Spain
jsantam@uca.es
[2] European Centre for Soft Computing, Edf. Científico Tecnológico, Mieres, Spain
[3] Dept. of Computer Science and A.I., University of Granada, Granada, Spain
oscar.cordon@softcomputing.es, ocordon@decsai.ugr.es
[4] Dept. of Software Engineering, University of Granada, Granada, Spain
sergio.damas@softcomputing.es, sdamas@ugr.es

Abstract. Photographic supra-projection is a complex and uncertain process that aims at identifying a person by overlaying a photograph and a model of the skull found. The more accurate the skull model is the more reliable the identification decision will be. Usually, forensics are obliged to perform a manual and time consuming process in order to obtain the model of the scanned forensic object. At least semiautomatic methods are demanded by these experts to assist them with this task. Our contribution aims to propose an evolutionary-based image registration methodology for the skull 3D model building problem. Experiments are performed over thirty two problem instances corresponding to a semi-automatic and fully automatic real skull model reconstruction.

1 Introduction

Photographic supra-projection [1] is a forensic process where photographs or video shots of the missing person are compared with the skull that is found. By projecting both photographs on top of each other (or, even better, matching a scanned three-dimensional skull model against the face photo/video shot), the expert can try to establish whether that is the same person. In this paper we will focus our attention on the first stage of the process: the accurate construction of a virtual model of the skull. Specifically, our concern is its frontal part to be able to perform the craniofacial identification. Our main goal will be to provide forensics with an automatic and accurate alignment methodology.

There is a need to use a range scanner to develop a computerized study of the skull. Multiple scans from different views are required to supply the information needed to construct the 3D model. The more accurate the alignment of the views (range images) the better the reconstruction of the object. Therefore, it is

* This work is supported by the Ministerio de Educación y Ciencia (ref. TIN2006-00829), including EDRF fundings.

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 415–422, 2007.
© Springer-Verlag Berlin Heidelberg 2007

fundamental to adopt a proper and robust technique to align the views in a common coordinate frame by means of *range image registration* (RIR) techniques, to avoid model distortion in the subsequent surface reconstruction stage [2].

The 3D skull model building problem when no positioning device is available is so complicated that it is one of the most time consuming tasks for the forensic experts. Therefore, software tools for the automation of their work are a real need. We will address the problem by means of a generic methodology based on *evolutionary algorithms* (EAs) for both the automatic and semiautomatic pair-wise RIR pre-alignment of several range images acquired from skulls. This proposal extends our previous contribution [3], where a *Scatter Search* (SS) EA [4] for RIR was introduced and applied to the current task. In this case, we propose a generic methodology making possible a fully automatic or at least semiautomatic approach, to help forensic experts in the first step towards the identification of a missing person. To do so, we will consider our proposal as well as other two recent EAs [5,6] within it.

The paper structure is as follows. In Section 2 we introduce the basis of our evolutionary-based methodology for the semiautomatic and fully automatic skull 3D model building. This section also presents the different evolutionary RIR techniques considered. Their suitability to tackle pair-wise RIR for 3D forensic model reconstruction is tested in Section 3 over different scenarios of skull modeling. Finally, in Section 4 we present some conclusions and future works.

2 Semiautomatic vs. Automatic Approaches for 3D Skull Model Reconstruction in Forensic Identification

Usually, pair-wise RIR methods are variants of the commonly known *Iterative Closest Point* (ICP) algorithm [7]. Since this algorithm requires a very small misalignment between the views, which is not always the case, most of the pair-wise RIR methods operate as follows [3,8]. First, a *pre-alignment* transformation or coarse alignment (a good approximation of the real one) is searched for typically by using an IR algorithm as a global search method. Then, a final *refinement* is applied, now as a local search process, typically an ICP-based method.

Depending on the range scanner precision, every view usually comprises thousands of 3D points. Since these views will be the input to our RIR problem, its complexity will depend on their sizes. If we are able to (semi)automatically synthesize these sets of points into a reduced version of them, we would both simplify the forensic expert task as well as reduce the problem complexity. To do so, we propose two different choices, according to a semiautomatic or a fully automatic RIR approach, respectively. On the one hand,*crest line extraction* [9]: where the point selection is based on the curvature of the 3D surfaces we aim at registering. On the other hand, *random sampling*, i.e., the uniform random selection of points along every 3D surface of every view.

Crest line extraction is not a trivial task. It requires the expertise of the user (in our case, the forensic expert, who has no knowledge in computer vision)

to perform a fine tuning of parameters to select only the more suitable points. However, more accurate results are expected with it, so our final aim is to be able to automatically generate a 3D skull model of enough quality without requiring the forensic expert to extract the crest lines (fully automatic approach).

There have been many proposals of pre-alignment methods that can provide a good starting point without requiring an initial guess [8]. To achieve the latter aim, our methodology will study the robustness of several evolutionary pre-alignment RIR contributions by evaluating the quality of their outcomes as proper initial estimations for the subsequent refinement step in our complex forensic anthropology scenario. For the semiautomatic approach, we have performed a crest lines extraction process by means of the Yoshizawa et al.'s proposal [9], and the points composing them are the only ones considered for the pre-alignment of the views. Then, the refinement step is applied to the original images. Any of the ICP proposals in the literature can be considered. After a preliminary experimentation, we selected Zhang's ICP-based proposal [7].

Three of the most accurate and recent evolutionary proposals in pair-wise RIR will be used used within our methodology. The reader is referred to the original contributions for more information:

- **Lomonosov et al.'s proposal:** it is focused on the use of a integer-coded *genetic algorithm* (GA) for the RIR pre-alignment problem [6].
- **Chow et al.'s proposal:** the use of a real-coded GA with suitable components is considered, like a sophisticated restart mechanism [5].
- **Santamaria et al.'s proposal:** our approach [3] is based on the adaptation of our previous IR proposal for medical MRIs [10] to apply SS to RIR.

3 Experimental Study

In this section we aim to analyze automatic and semiautomatic evolutionary approaches to generate 3D skull models of forensic objects. We will tackle the different problems the forensic expert has to deal with during the reconstruction stage of the photographic supra-projection process. Next, Section 3.1 describes the considered dataset. Sections 3.2 and 3.3 detail the experimental design and the parameter settings. Finally, Section 3.4 is devoted to the analysis of results.

3.1 Input Range Images

The Physical Anthropology Lab of the University of Granada provided us with a dataset of human skulls acquired by a Konica-Minolta© 3D Lasserscanner VI-910. We focused our study on the range images of a person who donated his remains for scientific proposes.

We have taken into account important factors along the scanning process like time and storage demand. Following the suggestions in [2], we considered a scan

every 45°. Hence, we deal with a sequence of eight different views: $0° - 45° - 90° - 135° - 180° - 225° - 270° - 315°$. The dataset we will use in our experiments is only limited to five of the eight views: $270° - 315° - 0° - 45° - 90°$. The reason is that our aim is to achieve a 3D model of the most interesting parts of the skull for the forensic expert and for the final objective of our work: the forensic identification of a missing person by photographic supra-projection.

3.2 Experimental Design

We will focus our attention on the design of pre-alignment methods as accurate, robust, and automatic as possible, especially when there is no positional device available. We will now propose a set of RIR problem instances with this aim. They will simulate an unsupervised scanning process, i.e. not oriented by any device. Likewise, we will be able to evaluate the performance of every semiautomatic and fully automatic RIR method considered as a pre-alignment technique.

For the semiautomatic approach, Yoshizawa et al.'s proposal [9] was considered to extract the crest lines. The original $270° - 315° - 0° - 45° - 90°$ views comprise 109936, 76794, 68751, 91590, and 104441 points, respectively. After crest lines extraction, the reduction in size is important: 1380, 1181, 986, 1322, and 1363 points. Figure 1 shows the correspondence between crestlines and the most prominent parts of the skull in every view. In the automatic RIR approach, we have followed an uniform random sampling of the input images. Hence, the only parameter the forensic expert must consider is the density of this random sampling. We fixed a 15% of the original dataset as a suitable value for the time and accuracy tradeoff.

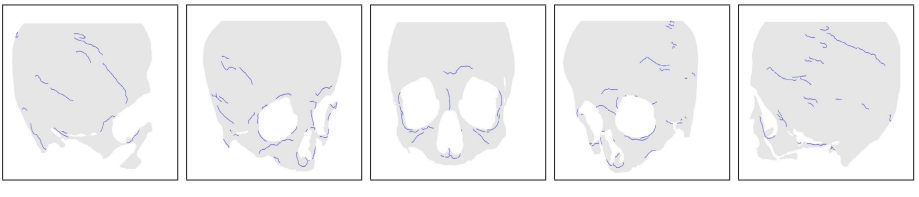


Fig. 1. From left to right: partial views of the skull and their corresponding crest lines acquired at $270°, 315°, 0°, 45°$, and $90°$, respectively

Specifically, we will consider four different rigid transformations. They are shown in Table 1 and will represent a typical bad initialization of the pre-alignment step by a forensic expert. Therefore, we are simulating the worst starting scenario. Any method that aims to become considered a good pre-alignment RIR technique will have to overcome such a bad initialization properly. These four transformations are applied to every adjacent pair of images of the 3D skull model leading to a global set of sixteen pair-wise RIR problem instances to be solved. Therefore, every method will finally deal with thirty two RIR problem instances (sixteen for every RIR approach: semi and fully automatic).

Table 1. Four rigid transformations considered

	θ	$Axis_x$	$Axis_y$	$Axis_z$	t_x	t_y	t_z
T_1	115.0°	-0.863868	0.259161	0.431934	-26.0	-15.5	-4.6
T_2	168.0°	0.676716	-0.290021	0.676716	6.0	5.5	-4.6
T_3	235.0°	-0.303046	-0.808122	0.505076	16.0	-5.5	-4.6
T_4	276.9°	-0.872872	0.436436	-0.218218	-12.0	5.5	-24.6

3.3 Parameter Settings

As we previously said, three recent contributions have been considered for our experimentation: GA_{Chow}, GA_{Lom}, and SS. Note that the first two methods perform a random sampling of several hundreds of points of the input images, in both automatic or semiautomatic approaches. Likewise, the objective function introduced in the previous proposals [5,6,3] must be slightly changed to be adapted to the strong difficulties we are imposing in this particular application, so that it considers the *Median Square Error (MedSE)* to deal with the small overlapping between adjacent views (see Figure 1):

$$MIN \quad F(f, I_s, I_m) \; = \; median\{\| \, f(\boldsymbol{p}_i) - \boldsymbol{p}_j{}' \, \|^2\}, \quad \forall \boldsymbol{p}_i \in I_s \tag{1}$$

where f is the transformation we are searching for, I_s and I_m are the input scene and model views, and $\boldsymbol{p}_j{}'$ is the closest model point to $f(\boldsymbol{p}_i)$ of the scene.

All the methods are run on a PC with an Intel Pentium IV 2.6 MHz. processor. In order to avoid execution dependence, fifteen different runs of each method in each pair-wise RIR problem instance have been performed.

We set all the parameter values in the same way that authors propose in their contributions [5,6,3]. Besides, the execution time for every pre-alignment method will be 20 and 100 seconds for the semiautomatic and fully automatic RIR approaches, respectively. The stop criterion for ICP_{Zhang} refinement stage was a maximum number of 200 iterations, which proved to be high enough to guarantee the convergence of the algorithm.

3.4 Analysis of Results

We have used the rotary stage as a positional device to actually validate the results from every RIR method. Since a high quality pre-alignment is provided from the scanner's software, a 3D model is available and it can be considered as the ground truth for the problem. Therefore, we know the global optimum location of every point in advance by using this 3D model. Unlike for the objective function, we will use the usual *Mean Square Error* (MSE) to measure the quality of the process, once the RIR method is finished. MSE is given by: $MSE = \sum_{i=1}^{r} ||f(\boldsymbol{x}_i) - \boldsymbol{x}_i{}'||^2/r$, where $f(\boldsymbol{x}_i)$ corresponds to the i-th scene point transformed by f (which is the result of every RIR method), r is the number of points in the scene image and $\boldsymbol{x}_i{}'$ is the same scene point but now using the optimal transformation f^* obtained by the positional device. Therefore, both $\boldsymbol{x}_i$

Table 2. Minimum (m), maximum (M), mean (μ), and standard deviation (σ) MSE values when tackling every problem instance (T_1, T_2, T_3, and T_4) related to the four pair-wise RIR scenarios from the **semiautomatic** approach. The best minimum (m) and mean (μ) values are highlighted using bold font

Semiautomatic RIR			T_1				T_2			
	I_{0^o} Vs. $T_i(I_{45^o})$		m	M	μ	σ	m	M	μ	σ
		SS	3.51	22.44	**11.73**	6.06	3.31	31.99	**13.54**	8.82
		$GA_{Lom.}$	1.48	5601.82	1032.79	1447.07	**1.31**	223.30	21.18	54.72
		GA_{Chow}	**1.31**	4518.76	2069.30	1858.09	1.39	4434.18	1857.96	1741.17
			T_3				T_4			
			m	M	μ	σ	m	M	μ	σ
		SS	3.41	17.30	**8.38**	5.07	3.12	13.35	**8.17**	2.69
		$GA_{Lom.}$	**1.38**	340.58	68.77	116.13	1.48	2672.86	391.27	699.74
		GA_{Chow}	2.21	5238.53	1158.02	1596.66	**1.40**	5368.14	1648.35	1526.32
	I_{0^o} Vs. $T_i(I_{315^o})$		T_1				T_2			
			m	M	μ	σ	m	M	μ	σ
		SS	**0.01**	0.01	**0.01**	0.00	**0.01**	0.01	**0.01**	0.00
		$GA_{Lom.}$	**0.01**	0.01	**0.01**	0.00	**0.01**	0.01	**0.01**	0.00
		GA_{Chow}	**0.01**	47.14	3.47	11.73	**0.01**	1084.44	79.20	269.39
			T_3				T_4			
			m	M	μ	σ	m	M	μ	σ
		SS	**0.01**	0.01	**0.01**	0.00	**0.01**	0.01	**0.01**	0.00
		$GA_{Lom.}$	**0.01**	0.03	**0.01**	0.01	**0.01**	0.01	**0.01**	0.00
		GA_{Chow}	**0.01**	43.02	4.24	11.39	**0.01**	2.07	0.16	0.51
	I_{45^o} Vs. $T_i(I_{90^o})$		T_1				T_2			
			m	M	μ	σ	m	M	μ	σ
		SS	**1.30**	1.42	**1.35**	0.04	1.32	1.41	**1.35**	0.02
		$GA_{Lom.}$	1.38	4945.76	331.77	1233.14	**1.26**	149.51	35.98	56.53
		GA_{Chow}	1.34	11348.23	801.60	2818.97	1.68	18764.62	3142.41	5722.17
			T_3				T_4			
			m	M	μ	σ	m	M	μ	σ
		SS	**1.30**	1.35	**1.32**	0.01	1.32	1.43	**1.36**	0.03
		$GA_{Lom.}$	1.39	18715.29	2513.49	6353.48	**1.26**	8708.39	582.79	2171.66
		GA_{Chow}	3.26	9569.70	805.29	2350.93	1.65	6234.63	485.92	1538.29
	I_{315^o} Vs. $T_i(I_{270^o})$		T_1				T_2			
			m	M	μ	σ	m	M	μ	σ
		SS	3.61	3.63	**3.62**	0.01	2.89	3.62	3.33	0.34
		$GA_{Lom.}$	**1.75**	11143.11	749.93	2777.70	**1.86**	4.78	**3.01**	0.78
		GA_{Chow}	2.45	5498.43	683.14	1624.90	2.34	10860.12	4238.43	5154.46
			T_3				T_4			
			m	M	μ	σ	m	M	μ	σ
		SS	2.73	3.62	**3.29**	0.36	3.62	3.62	3.62	0.00
		$GA_{Lom.}$	**1.50**	29.13	8.81	9.17	**1.73**	3.62	**2.65**	0.62
		GA_{Chow}	2.48	10777.19	2210.65	3805.84	1.87	15.55	5.62	4.25

and $\boldsymbol{x}_i{}'$ are the same point but its location can differ if $f \neq f^*$. Our aim with this MSE definition is to take advantage of the availability of an *a priori* optimum model to study the behavior of the RIR methods. Indeed, this evaluation is not applicable in real environments where no optimum model is available.

Tables 2 and 3 show the results of the semiautomatic and automatic approach, respectively. The first conclusion is the good performance of all the analyzed evolutionary proposals. Indeed, most of the minimum values (m) of all the methods in both approaches are close to zero in almost every problem instance. These results reinforce our evolutionary methodology to the pair-wise RIR in this forensic problem. On the other hand, if we compare every result in Table 2 and in Table 3 we conclude that every method behaves in a more suitable way when dealing with the semiautomatic approach than when tackling the automatic one. As it was initially expected, the synthesis of data provided by the crest lines is very helpful for every method. However, as said, this preprocessing requires the expertise for the proper crest lines extraction. Fortunately, we can see how SS is able to provide a reliable reconstruction of the skull in the automatic approach, although it is the only of the three evolutionary methods considered able to achieve this goal.

Finally, Figure 2 aims to summarize the previous comments graphically. Since we are specially interested in the automatic approach and the difference in

Table 3. Results for the **automatic** approach

Automatic RIR

I_{0^o} Vs. $T_i(I_{45^o})$

	T_1				T_2			
	m	M	μ	σ	m	M	μ	σ
SS	**1.49**	11479.33	**7553.89**	2465.41	**1.37**	5630.40	**1417.78**	2351.80
$GA_{Lom.}$	3099.70	15578.35	8401.16	4976.56	854.61	6834.97	3513.20	1649.80
GA_{Chow}	1074.09	15143.81	8067.05	4094.66	1724.65	16093.26	5907.46	3754.55
	T_3				T_4			
	m	M	μ	σ	m	M	μ	σ
SS	**1.50**	3112.36	**1112.95**	1399.51	**1.48**	9919.53	**6585.65**	4092.68
$GA_{Lom.}$	4.53	3780.89	1684.41	1131.34	16.84	17851.20	7658.33	4701.49
GA_{Chow}	208.77	10927.11	3762.57	2495.64	13.59	18101.16	9717.35	4604.05

I_{0^o} Vs. $T_i(I_{315^o})$

	T_1				T_2			
	m	M	μ	σ	m	M	μ	σ
SS	**0.01**	0.01	**0.01**	0.00	**0.01**	0.01	**0.01**	0.00
$GA_{Lom.}$	**0.01**	82.40	8.16	21.05	**0.01**	6433.83	611.05	1689.94
GA_{Chow}	**0.01**	7162.80	1189.77	2225.74	**0.01**	21165.96	8258.95	7664.76
	T_3				T_4			
	m	M	μ	σ	m	M	μ	σ
SS	**0.01**	0.01	**0.01**	0.00	**0.01**	0.01	**0.01**	0.00
$GA_{Lom.}$	**0.01**	15292.79	8034.59	6503.39	**0.01**	6429.36	1273.88	2519.95
GA_{Chow}	**0.01**	17992.58	6658.78	7166.29	**0.01**	14673.30	1664.11	3612.26

I_{45^o} Vs. $T_i(I_{90^o})$

	T_1				T_2			
	m	M	μ	σ	m	M	μ	σ
SS	**1.24**	19518.23	7638.99	9355.58	**1.18**	18781.22	**1253.63**	4684.44
$GA_{Lom.}$	1.78	19420.03	**6452.13**	9058.42	1.34	19601.43	5172.05	8502.45
GA_{Chow}	2.24	19952.58	13358.92	8227.04	1.34	19816.55	7776.61	9176.55
	T_3				T_4			
	m	M	μ	σ	m	M	μ	σ
SS	**1.13**	19400.45	**2586.27**	6590.61	**1.10**	1.27	**1.24**	0.03
$GA_{Lom.}$	1.25	19411.10	3864.91	7705.66	1.31	18795.58	2518.45	6373.43
GA_{Chow}	3.61	19505.41	12540.78	8854.98	2.42	19768.89	11240.35	8760.94

I_{315^o} Vs. $T_i(I_{270^o})$

	T_1				T_2			
	m	M	μ	σ	m	M	μ	σ
SS	2.72	20375.45	**1361.59**	5081.66	2.72	21605.61	5742.59	9517.60
$GA_{Lom.}$	**2.45**	20000.45	2658.30	6767.95	2.61	20600.98	**1377.29**	5137.74
GA_{Chow}	2.65	20192.43	13260.02	8590.65	**2.57**	20139.24	10200.25	8796.53
	T_3				T_4			
	m	M	μ	σ	m	M	μ	σ
SS	2.81	21290.11	**2788.46**	7101.72	**2.02**	3.62	**3.20**	0.5
$GA_{Lom.}$	**2.47**	19951.30	3951.35	7893.31	2.29	4.17	3.42	0.5
GA_{Chow}	2.48	21362.86	9265.16	9920.66	2.60	21155.96	5491.44	9085.52

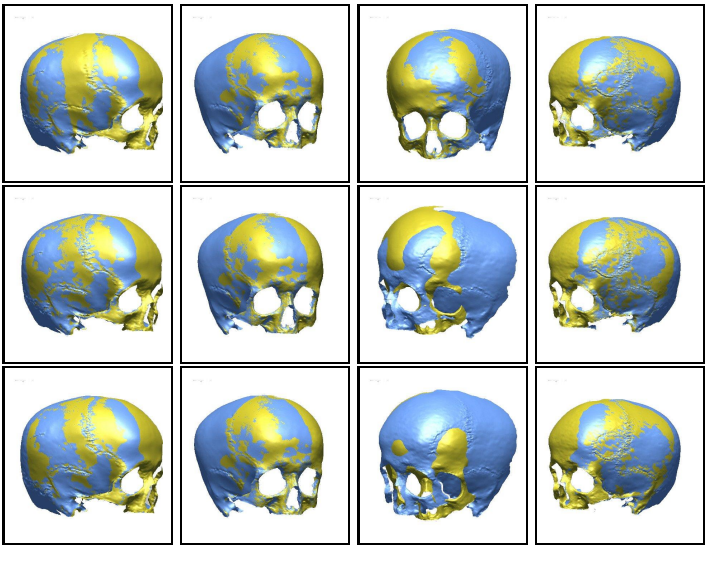


Fig. 2. From top to bottom: the worst minimum results (i.e. the worst m in every $\{T_1, T_2, T_3, T_4\}$ set) of the four pair-wise RIR scenarios corresponding to SS (first row), GA_{Lom} (second row), and GA_{Chow} (third row)

performance is more easily identified in it, we will focus on this approach. Specifically, we present the worst minimum results (i.e. the worst m in every $\{T_1, T_2, T_3, T_4\}$ set) of the four pair-wise RIR scenarios corresponding to SS (first row), GA_{Lom} (second row), and GA_{Chow} (third row). Such worst-case visualization will stress the difference in performance among the proposals of this methodology. In particular, note the difficulties when tackling the $I_{0^o}\ Vs.\ T_i(I_{45^o})$ scenario corresponding to the third column of the figure except for the SS method (first row).

4 Concluding Remarks

We have detailed the suitability of range scanners for the reconstruction of reliable models in the forensic photographic supra-projection process. There are scenarios where a positional device which automatically builds the model cannot be used. Moreover, the latter devices often fail when solving the problem. We have proposed a semiautomatic/automatic evolutionary methodology to solve the previous problems and we have analyzed three recent EAs as pair-wise RIR methods [5,6,3] within it. From the results obtained, we have demonstrated that a fully automatic approach is possible by using our SS-based proposal [3], outperforming the other two methods considered in terms of performance and robustness. We are planning to extend this study to other pre-alignment methods [8] within the proposed methodology.

References

1. M. Y. Iscan: Introduction to techniques for photographic comparison. In M. Y. Iscan and R. Helmer, Eds., Forensic Analysis of the Skull, Wiley, 57–70, 1993.
2. L. Silva, O. Bellon, and K. Boyer: Robust range image registration using genetic algorithms and the surface interpenetration measure. World Scientific, 2005.
3. J. Santamaría, O. Cordón, S. Damas, I. Alemán, and M. Botella: A Scatter Search-based technique for pair-wise 3D range image registration in forensic anthropology. Soft Computing, In press, 2007.
4. M. Laguna and R. Martí: Scatter search: methodology and implementations in C. Kluwer Academic Publishers, 2003.
5. C. K. Chow, H.T. Tsui, and T. Lee: Surface registration using a dynamic genetic algorithm. Pattern Recognition, 37: 105–117, 2004.
6. E. Lomonosov, D. Chetverikov, and A. Ekart: Pre-registration of arbitrarily oriented 3D surfaces using a GA. Pattern Rec. Letters 27(11): 1201–1208, 2006.
7. Z. Zhang: Iterative point matching for registration of free-form curves and surfaces. International Journal of Computer Vision, 13(2): 119–152, 1994.
8. J. Salvi, C. Matabosch, D. Fofi, and J. Forest: A review of recent range image registration methods with accuracy evaluation. Image Vision Comp., In press, 2007.
9. S. Yoshizawa, A. Belyaev, and H. Seidel: Fast and robust detection of crest lines on meshes. Proc. 2005 ACM Symp. on Solid and Physical Modeling, 227–232, 2005.
10. O. Cordón, S. Damas, and J. Santamaría: A fast and accurate approach for 3D image registration using the scatter search evolutionary algorithm. Pattern Recognition Letters, 27(11): 1191–1200, 2006.

Scale Invariance for Evolved Interest Operators

Leonardo Trujillo and Gustavo Olague

Proyecto Evovisión,
Centro de Investigación Científica y de Educación Superior de Ensenada, Km. 107
Carretera Tijuana-Ensenada, 22860, Ensenada, BC, México
trujillo@cicese.mx, olague@cicese.mx
http://ciencomp.cicese.mx/evovision

Abstract. This work presents scale invariant region detectors that apply evolved operators to extract an interest measure. We evaluate operators using their repeatability rate, and have experimentally identified a plateau of local optima within a space of possible interest operators Ω. The space Ω contains operators constructed with Gaussian derivatives and standard arithmetic operations. From this set of local extrema, we have chosen two operators, obtained by searching within Ω using Genetic Programming, that are optimized for high repeatability and global separability when imaging conditions are modified by a known transformation. Then, by embedding the operators into the linear scale space generated with a Gaussian kernel we can characterize scale invariant features by detecting extrema within the scale space response of each operator. Our scale invariant region detectors exhibit a high performance when compared with state-of-the-art techniques on standard tests.

1 Background

Current trends in Computer Vision (CV) are to adopt a simplified approach to address the problems of: object detection/recognition, content based image retrieval and image indexing [1]. This approach works with image information extracted directly from local image features which makes it robust to partial object occlusions and eliminates the need for prior segmentation. However, the approach does require the detection of stable image features that correspond to *visually interesting regions*, of which interest points are the most widely known [2,3,4]. Interest regions exhibit a high level of variation with respect to a local measure that is extracted using a particular image operator. Hence, different region detectors define different operators that extract an interest measure for every image pixel. After applying the interest operator local extrema are selected as interest regions. The main characteristic, and the only one for which a reliable performance metric exists, that is expected from an interest region operator is stability under changes in viewing conditions quantified by the repeatability rate [2]. Stability is evaluated under different kinds of transformations, which include: *translation, illumination change, rotation, scale change and projective transformations*. Interest region detectors invariant to the first three types of

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 423–430, 2007.
© Springer-Verlag Berlin Heidelberg 2007

image transformations are better know as interest point detectors [3,4], invariance to the first four are scale invariant region detectors [5], while invariance to all are know as affine covariant region detectors [5]. Previous work by Trujillo and Olague [3,4] proposed a novel approach to construct optimized interest point detectors using Genetic Programming (GP) as an optimization engine and the repeatability rate as part of the fitness. The present work extends that contribution by embedding evolved operators into a linear scale space to detect scale invariant regions [6], and presents the following contributions to the field of automatic feature extraction with Evolutionary Computation. First, this work characterizes a conceptual search space for interest operators, that are applicable to different types of CV applications. Second, we identify how artificial evolution automatically rediscovered basic image analysis techniques that have long been considered as possible models for low level vision in biological visual systems. Finally, our scale invariant detectors that are based on evolved operators achieve better performance than manmade designs, and have a simpler structure.

Interest Operators. These are functions that operate on a local neighborhood of every image pixel and extract a corresponding interest measure K, thereby producing an *interest image* which is thresholded to detect extrema. Popular interest operators, designed to detect interest points, include [7,8]:

$$K_{Harris\&Stephens}(\mathbf{x}) = det(A) - k \cdot Tr(A)^2 \ ,$$

$$K_{Forstner}(\mathbf{x}) = \frac{det(A)}{Tr(A)} \ ,$$

where A is the local autocorrelation matrix [5] given by

$$A(\mathbf{x}, \sigma_I, \sigma_D) = \sigma_D^2 \cdot G_{\sigma_I} * \begin{bmatrix} L_x^2(\mathbf{x}, \sigma_D) & L_x L_y(\mathbf{x}, \sigma_D) \\ L_x L_y(\mathbf{x}, \sigma_D) & L_y^2(\mathbf{x}, \sigma_D) \end{bmatrix} \ ,$$

σ_D and σ_I are the derivation and integration scales respectively, L_u is the Gaussian derivative in direction u and G_σ is a Gaussian smoothing function with standard deviation σ. Other interest measures are related to the curvature at each point, such as the determinant of the Hessian proposed by Beaudet [9]:

$$K_{Beaudet}(\mathbf{x}) = I_{xx}(\mathbf{x}, \sigma_D) \cdot I_{yy}(\mathbf{x}, \sigma_D) - I_{xy}^2(\mathbf{x}, \sigma_D) \ .$$

Wang and Brady [10] propose an interest measure related to the curvature of an edge using the Laplacian along with the gradient magnitude,

$$K_{Wang\&Brady}(\mathbf{x}) = (\nabla^2 I)^2 - s|\nabla I|^2 \ .$$

Constructing Interest Operators with Genetic Programming. Early contributions to this problem include [11], however previous works do not define a proper fitness measure and their results are neither reusable or general. A novel framework to automatically synthesize interest operators with GP, that overcomes the shortcomings of [11], was presented in [3,4]. From a careful analysis of the above mentioned operators, as well as others, the authors define the following

Function and Terminal sets that would allow us to construct any of them, as well as a vast amount of unknown operators,

$$F = \left\{ +, -, |-|, *, /, I^2, \sqrt{I}, log_2, \frac{I}{2}, G_{\sigma=1}, G_{\sigma=2}, EQ(I) \right\} \ , \tag{1}$$

$$T = \{ I, L_x, L_{xx}, L_{xy}, L_{yy}, L_y \} \ , \tag{2}$$

where F and T are the function and terminal sets respectively, and $EQ(I)$ is an histogram equalization. Some authors [8,10,9] do not use Gaussian derivatives, however T is defined in this way because they are less susceptible to noise. On the other hand, an appropriate evaluation function $f(o)$ should depend on the repeatability rate $r_{o,J}(\epsilon)$ of each operator o on an image sequence J. Therefore, the fitness is $f(o) \propto r_{o,J}(\epsilon)$, where ϵ is an error threshold, see [2,3,4].

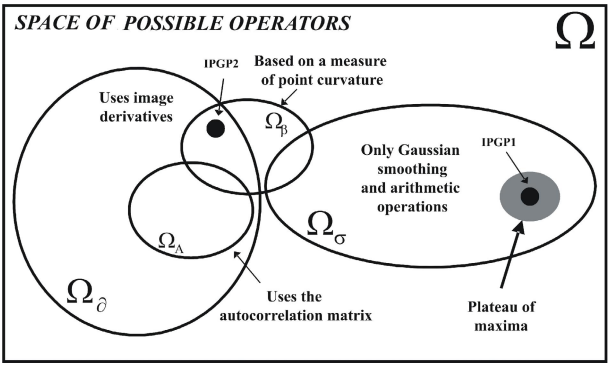


Fig. 1. Space of possible interest operators Ω with the evolved operators $IPGP1$ and $IPGP2$ presented in [3,4] and their position within Ω

Figure 1 represents a high level view of the space Ω of possible interest operators constructed with the above mentioned primitives. A subspace $\Omega_\delta \subset \Omega$ represents the space of possible operators that use image derivatives explicitly, taken from T, to obtain their interest measure; $\Omega_\sigma \subset \Omega$ only contains operators that use Gaussian smoothing and arithmetic operations included in F. The subspaces Ω_δ and Ω_σ group operators based on their genotype and not their phenotype. Figure 1 also shows where we might find the subspace of operators that rely on measures pertaining to the local autocorrelation matrix Ω_A, or that extract a measure related to surface curvature Ω_β[1], along with the two operators presented in [3,4], $IPGP1$ and $IPGP2$ [2]. These operators outperformed or matched previous manmade designs on standard tests [14],

$$K_{IPGP1}(\mathbf{x}) = G_{\sigma=2} * (G_{\sigma=1} * I - I) \ , \tag{3}$$

[1] Ω_β contains operators with similar functionality and its intersection with other subspaces, those based on structure, may or may not be empty.

[2] $IPGP$ is an acronym for *Interest Point Detection with GP*.

$$K_{IPGP2}(\mathbf{x}) = G_{\sigma=1} * (L_{xx}(\mathbf{x}, \sigma_D = 1) \cdot L_{yy}(\mathbf{x}, \sigma_D = 1) - L^2_{xy}(\mathbf{x}, \sigma_D = 1)) \ . \quad (4)$$

$IPGP1$ identifies salient low intensity features, while its additive inverse extracts high intensity features. $IPGP2$ on the other hand is a modified version of the operator proposed by Beaudet [9], similar to the improvements made in [2] to the Harris and Stephens detector, what the authors called *Improved Harris*. Further experimental runs of the GP search have identified a plateau of local maxima in the neighborhood of $IPGP1$. Here, we present a close neighbor, both in the function space and fitness space, of $IPGP1$ that we name $IPGP1^*$,

$$K_{IPGP1^*}(\mathbf{x}) = G_{\sigma=2} * |G_{\sigma=2} * I - I| \ . \quad (5)$$

$IPGP1^*$ identifies maxima related to both $IPGP1$ and its additive inverse.

Proposition 1. *Both $IPGP1$ and $IPGP1^*$ are proportional to DoG (Difference-off-Gaussian) filters, if we assume that image I is derived from an unknown image $\hat{I}$ blurred with a Gaussian of unknown standard deviation $\hat{\sigma}$ such that $I = G_{\hat{\sigma}} * \hat{I}$, and*

$$G_\sigma * I - I = G_\sigma * G_{\hat{\sigma}} * \hat{I} - G_{\hat{\sigma}} * \hat{I} \propto G_{\sigma+\hat{\sigma}} * \hat{I} - G_{\hat{\sigma}} * \hat{I} = (G_{\sigma+\hat{\sigma}} - G_{\hat{\sigma}}) * \hat{I} \ . \quad (6)$$

Therefore, $IPGP1$ and $IPGP1^*$ are approximations of the 2D LoG function.

2 Scale Space Analysis

One of the basic problems in CV is to determine the scale at which image information should be analyzed. Different real world structures are only appreciable and relevant at certain scales and lack importance at others. Thus, a solution to this problem has been proposed by applying the concept of scale-space, which allows us to work explicitly with the scale selection problem while also simplifying image analysis by only focusing on *interesting* scales. For a useful, if not rigorous, concept of scale we turn to one of the most important contributions in scale-space theory by Lindeberg [6].

"The scale parameter should be interpreted only as an abstract scale parameter implying a weak ordering property of objects of different size without any direct mapping from its actual value to the size of features in a signal represented at that scale".

A multi-scale representation of an image is obtained by embedding it within a family of derived signals which depend on the lone scale parameter t [6].

Definition 1. *Given a signal $f : \Re^D \to \Re$, the linear scale-space representation $L : \Re^D \times \Re \to \Re$ of f is given by the solution to the diffusion equation*

$$\delta_t L = \frac{1}{2}\nabla^2 L = \frac{1}{2}\sum_{i=1}^{D} \delta_{x_i x_i} L \ , \quad (7)$$

with the initial condition $L(\cdot; 0) = f(\cdot)$, for which the Green function solution is the Gaussian kernel. Equivalently, it is possible to define the scale-space as the

family of derived signals obtained by convolving a signal f with Gaussian filters at different scales t (standard deviation),

$$L(\cdot;t) = G_t * f(\cdot) \ . \tag{8}$$

Lindeberg notes that the scale-space-representation could be taken as a canonical model for biological vision due to results in neurophysiological studies [12]. Now, to determine the scale at which image features should be analyzed Lindeberg presents a **Principle for scale selection** [13]: *"In the absence of other evidence, assume that a scale level, at which some (possibly non-linear) combination of normalized derivatives assumes a local maximum over scales, can be treated as reflecting a characteristic length of a corresponding structure in the data"*. Normalized derivatives are invariant at different scales [6]. In practice however, Lindeberg concludes that the usefulness of the principal for scale selection *"... must be verified empirically, and with respect to the type of problem it is to be applied to"*. Hence, we can expect that an experimental GP search for candidate interest operators is a valid approach to construct a scale invariant detector based on a *"possibly non-linear combination of normalized derivatives"*. Furthermore, it is possible to contemplate that the GP search will be biased to simplified approximate measures, such as the approximation of the LoG by way of DoG filters. This can be induced in a GP search by applying specific genetic or selection operators that help keep evolved operators simple [3,4].

Characteristic Scale. From an algorithmic point of view, selecting a characteristic scale for local image features is a process in which local extrema of a function response are found over different scales [13]. Given a function, or interest operator, $F(\mathbf{x};t_i)$ that computes an interest measure for each image pixel $\mathbf{x}$ at different scales t_i, we can assume that the characteristic scale at $\mathbf{x}$ is t_n if

$$F(\mathbf{x};t_n) > \sup\{F(\mathbf{x_W};t_n), F(\mathbf{x_W};t_{n-1})F(\mathbf{x_W};t_{n+1}) | \forall \mathbf{x_W} \in \mathbf{W}, \mathbf{x_W} \neq \mathbf{x}\}$$
$$\wedge F(\mathbf{x};t_n) > h \ , \tag{9}$$

where h is a threshold, and $\mathbf{W}$ is a $n \times n$ neighborhood around $\mathbf{x}$. This process, similar to what is done for interest point detection, will return a set of local scale invariant regions, each centered on an image pixel $\mathbf{x}$.

3 Scale Invariant Detectors

Now that we have defined the concept of characteristic scale and a basic methodology on how to obtain it, we can move on to present our proposed scale invariant detectors. Here, we are interested in detectors derived from the scale-space representation. As a starting point, we turn to Mikolajczyk and Schmid [5] who gave a comparison of different scale invariant detectors, including: DoG [15], Hessian, Laplacian [13] and Harris-Laplace [5]. From this comparison the authors experimentally concluded that, as expected, the DoG and Laplacian gave very similar results and that the Harris-Laplace detector gave the highest repeatability rate for scale change transformations. As mentioned before, we will

present detectors based on the $IPGP1$ and $IPGP1^*$ interest operators, which according to proposition 1 are proportional to DoG filters. However, we present a different algorithmic implementation that maintains the basic structure of the operator and produces better performance. Scale invariant detection using DoG as proposed by Lowe [15] uses a scale space pyramid and DoG filters are applied between adjacent scales. Here, our $IPGP$ based detectors will perform DoG filtering between each scale and the base image, with scale $t = 0$, contrasting with the implementation in [15] in which both Gaussian functions of the DoG filter are modified sequentially. In order to apply our evolved operators within a scale-space analysis we must modify their definition by including the scale parameter,

$$K_{IPGP1_t}(\mathbf{x}; t) = G_{t_i} * (G_{t_i} * I - I) \ , \tag{10}$$

$$K_{IPGP1^*_t}(\mathbf{x}; t) = G_{t_i} * |G_{t_i} * I - I| \ , \tag{11}$$

where t_i is the current scale, $i = 1...N$ with N the total number of scales analyzed. Now our operators are scale dependent and will return a different interest measure for each pixel at different scales. Hence, it is now possible to apply the characteristic scale selection criteria. Our operators avoided the need for normalized derivatives and are more efficient than other detectors as reported in [5]. Note that we are not using operators evolved explicitly for high repeatability under scale change. However, current state-of-the-art detectors rely on interest point operators embedded into the linear scale-space, the same approach we are taking. This can be seen as having a relationship with the area of interactive evolution where user input guides the selection process.

Implementation. This step is straightforward, its only requirements are to establish a set of parameters that are defined empirically, as is the case for all region detectors. We set $N = 20$ and $t_i = 1.2^i$; the size of our scale neighborhoods

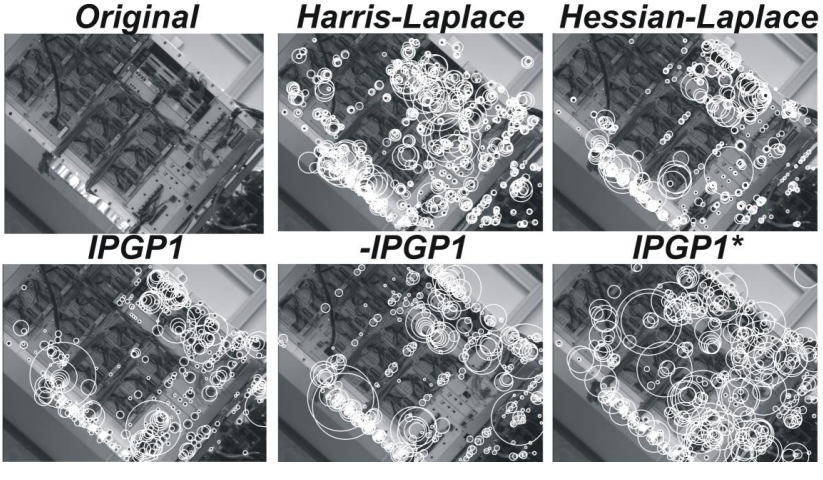


Fig. 2. Sample image regions. Regions are shown with circles of radius $r = 3 \cdot t_n$ pixels, with t_n the regions characteristic scale.

W was set to $n = 5$, our thresholds h are set to 10 for $IPGP1$ and its additive inverse, and 15 for $IPGP1^*$. For comparison purposes, we use the Harris-Laplace and Hessian-Laplace detectors, using the authors binaries downloaded from the Vision Geometry Group website [14], along with five image sequences: Laptop, BIP, VanGogh, Asterix and Boat; where the first four are sequences that only present scale change transformations while the fifth has both scale and rotation.

Results. Figure 2 is a qualitative comparison that shows interest regions extracted by each detector. It is possible to observe how $IPGP1$ and its additive inverse extract complementary regions, while $IPGP1^*$ extracts a combination of maxima from both. Furthermore, the $IPGP$ operators and the Harris and Hessian based methods exhibit similarities. Figure 3 is a quantitative comparison of each detector, it presents the repeatability rate on 5 different image sequences. The performance graphics plot the images in the sequence and the repeatability rate with respect to the base image. Each image in the sequence is progressively transformed, i.e., the view point of image 4 is closer to that of the base image than the viewpoint of image 10 in the BIP sequence [14]. All detectors exhibit similar performance patterns. However, we can appreciate that the detectors based on evolved operators are slightly better on most sequences.

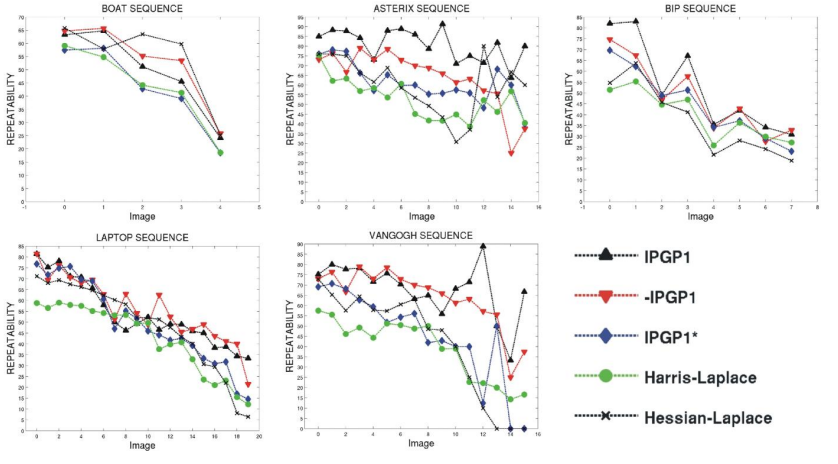


Fig. 3. Repeatability of each detector in our comparison, for each test sequence

4 Discussion and Future Work

This paper presented scale invariant detectors based on operators optimized for high repeatability using GP. The detectors were embedded into a linear scale space generated with a Gaussian kernel and compared with state-of-the-art detectors. Results show that our detectors are, on average, better than other detectors based on their repeatability rate, while at the same time maintaining a simpler structure. Our results show that simple operators found by simulated

evolution can outperform more elaborate manmade designs. This interesting result substantiates the belief that evolution will always find the simplest and most apt solution to a given problem. This is made possible by correctly framing the evolutionary search process with a fitness function that promotes the extraction of highly repeatable regions, a property that is useful in many vision applications. As possible future work, we suggest two main extensions. First, employ an evolutionary search process that directly takes into account scale space analysis in its fitness evaluation. Second, extend the use of evolved operators to extract affine covariant features, a much more challenging problem in CV.

Acknowledgments. Research funded by UC MEXUS-CONACyT Collaborative Research Grant 2005 through the project "Intelligent Robots for the Exploration of Dynamic Environments". First author supported by scholarship 174785 from CONACyT. This research was also supported by the LAFMI project.

References

1. K. Mikolajczyk and C. Schmid: A Performance Evaluation of Local Descriptors. IEEE Trans. Pattern Anal. Mach. Intell., **27**(10) (2005) 1615-1630.
2. C. Schmid, R. Mohr, and C. Bauckhage: Evaluation of interest point detectors. Int. J. Comput. Vision, **37**(2) (2000) 151–172.
3. L. Trujillo and G. Olague: Synthesis of interest point detectors through genetic programming, in Proceedings of GECCO'06, Vol.1, (ACM Press 2006), 887–894.
4. L. Trujillo and G. Olague: Using Evolution to Learn How to Perform Interest Point Detection, in Proceedings of ICPR'06, Vol. 1, (IEEE Comput. Soc. 2006), 211–214.
5. K. Mikolajczyk and C. Schmid: Scale and Affine Invariant Interest Point Detectors. Int. J. Comput. Vision, **60**(1) (2004) 63–86.
6. T. Lindeberg: Discrete Scale-Space Theory and the Scale-Space Primal Sketch. PhD thesis, Computational Vision and Active Perception Laboratory (CVAP) Royal Institute of Technology, Sweden, (1991).
7. C. Harris and M. Stephens: A combined corner and edge detector, in Alvey Vision Conference, (1988) 147–151.
8. W. Förstner: A framework for low level feature extraction, in Proceedings of the 3rd European Conference on Computer Vision, (1994) 383–394.
9. P. R. Beaudet: Rotational invariant image operators, in Proceedings of the 4th International Joint Conference on Pattern Recognition, (1978) 579–583.
10. H. Wang and J.M. Brady: Corner detection for 3d vision using array processors, in Proceedings from BARNAIMAGE-91, (Springer-Verlag 1991).
11. M. Ebner and A. Zell: Evolving Task Specific Image Operator, in Proceedings from EvoIASP'99 and EuroEcTel'99, Lecture Notes in Computer Science Vol. 1596, (Springer-Verlag 1999) 74–89.
12. R.A. Young: Simulation of the Human Retinal Function with the Gaussian Derivative Model, in Proceedings of the 1986 IEEE Conference on Computer Vision and Pattern Recognition, (1986) 564–569.
13. T. Lindeberg: Feature detection with automatic scale selection. Int. J. Comput. Vision, **30**(2) (1998) 79–116.
14. Visual Geometry Group: http://www.robots.ox.ac.uk/ vgg/research/
15. D.G. Lowe: Object recognition from local scale-invariant features, in Proceedings of CVPR 1999, Vol. 2, (IEEE Comput. Soc. 1999) 1150–1157.

Application of the Univariate Marginal Distribution Algorithm to Mixed Analogue - Digital Circuit Design and Optimisation

Lyudmila Zinchenko[1,2], Matthias Radecker[1], and Fabio Bisogno[1]

[1] FhG-IAIS Schloss Birlinghoven 53754 Sankt – Augustin Germany
{matthias.radecker,fabio.bisogno,
lyudmila.zinchenko}@iais.fraunhofer.de
[2] TNURE, l. Nekrasovsky, 44, Taganrog, 347928, Russia

Abstract. Design and optimisation of modern complex mixed analogue-digital circuits require new approaches to circuit sizing. In this paper, we present a novel approach based on the application of the univariate marginal distribution algorithm to circuit sizing at the system level. The results of automotive electronics circuits sizing indicate that all design requirements have been fulfilled in comparison with a human design. Experiments indicate that elitism increases the performance of the algorithm.

1 Introduction

The growing market of mixed analogue–digital circuits is observed in modern electronic systems for automotive applications, telecommunication systems, etc. High level of system integration and the rapid evolution of process technologies result in significant complication of design process. A key to manage design and circuit complexities is a wide application of computer aided design tools. CAD tools are effective in the automation of the routine and repetitive design tasks reducing design time and design cost. Different optimisation techniques are used in commercial tools (Neocircuit, Circuit Explorer, WiCkeD, etc.). However, a comparison of genetic algorithms and other methods for transistor sizing [1] has shown advantages of the genetic algorithms approach.

Moreover, the major bottleneck in design of mixed analogue-digital systems is design efficiency. The commercial tools size library cells (comparators, amplifiers, etc.). However, a design based on fully optimised library cells may fail to meet all design specifications at the system level. It results in additional iterations during design cycle and additional human efforts.

In this paper we present our results on an application of evolutionary probabilistic algorithms to design and optimisation of mixed digital-analogue circuits. We compare the effectiveness of classical univariate marginal distribution algorithm (UMDA) and its modification by elitism for static fitness schedule. We use symmetry recognition circuit as our benchmark. This circuit is used in automotive control system based on piezoelectric transformer application [2].

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 431–438, 2007.
© Springer-Verlag Berlin Heidelberg 2007

The remainder of the paper is organized as follows. Section 2 briefly summaries results in automated circuit design and optimisation. Section 3 elaborates on the features of our mixed analogue-digital circuit design and optimisation process. The experimental settings and results are presented in Sections 4 and 5. We discuss the conclusions in Section 6.

2 Approaches to Automated Circuit Design and Optimisation

Circuit sizing is an important step in mixed analogue-digital circuit design process. It aims to find circuit parameters such that design characteristics (performance, area, power consumption, etc.) are optimised. There are two main approaches to size analogue circuits based on the use of expert knowledge and on optimisation techniques [3]. However, knowledge–based analogue sizing approaches are complex for formalization and inflexible [3]. Therefore, they failed in industrial applications.

Different optimisation techniques are widely used in circuit optimisation for industrial application. Several experimental tools for analogue circuit optimisation have been developed (OPTIMAN [4], MAELSTROM [5], ANACONDA [6], ASTRX/OBLX [7], ASF [8], etc.). They differ by the circuit performance evaluations and by the search algorithm. OPTIMAN [4] uses analytical models that are fast, but their accuracy is better for small-signal characteristics. Another approach is based on numerical simulation in the loop of the optimisation. MAELSTROM [5] and ANACONDA [6] use SPICE, while ASTRX/OBLX [7] exploits the asymptotic evaluation. ASF [8] combines SPICE and the analytical approaches. Simulated annealing and its different modifications are mainly used in the tools mentioned above [4, 7, 8]. A combination of genetic algorithms and simulated annealing in MAELSTROM [5] allows to enhance design capabilities. A novel algorithm based on combination of evolutionary strategies and simulated annealing optimises analogue circuit characteristics [9]. The approach [5] has been expanded for mixed signal circuits in the commercial tool NeoCircuit. Another commercial tool leveraging intelligent systems techniques supports multiobjective optimisation formulation for mixed-signal circuits (AMS Genius at Analog Design Automation, now acquired by Synopsys). The commercial tool WiCkeD [10] is based on the application of numerical optimisation algorithms for mixed signal circuits. However, these tools are available for optimisation at the cell level only.

An increasing complexity of electronic systems requires a design and optimisation methodology at higher hierarchical design levels. Up to date, there is no method to resolve the problem if the system does not meet all design specifications [11]. In the worst case, the complete redesign may be required. It increases the time to market and the development cost.

A design of a system with hundreds of transistors is too complex to attack at once. A long simulation run-time is another crucial obstacle. Smart algorithms are required to manage the complexity of the problem. They should be able to find a good solution with reasonable computational costs.

Our focus is on the application of evolutionary probabilistic models (EPM). They have been recognized as a new computing paradigm in evolutionary computation. EPM include the estimation of distribution algorithms, probabilistic model building

genetic algorithms, ant colony optimisation, cross entropy methods [13]. There is no traditional genetic algorithms crossover or mutation in evolutionary probabilistic algorithms. Instead, they explicitly extract global statistical information from their previous search and build a probability distribution model of promising solutions, based on the extracted information. New solutions are sampled from the probabilistic model. Probabilistic evolutionary algorithms represent a new systematic way to solve hard search and optimisation problems. They have shown to resolve a number of problems the conventional genetic algorithms experience great difficulties with and solve a number of difficult problems quickly, accurately, and reliably [13].

In [12] Mühlenbein showed that genetic algorithms can be approximated by an algorithm using univariate marginal distributions only. UMDA is an evolutionary algorithm, which combines mutation and recombination by means of distribution. The distribution is estimated from a set of selected points. It is used then to generate new points for the next generation. In order to improve design abilities, mutation has been introduced into UMDA by a concept called Bayesian prior [13]. UMDA with Bayesian prior is able to overcome local minima. Furthermore, we used the experiments to make a reasonable choice of Bayesian prior for analogue circuit design [14] and an effective circuit representation [15]. In paper we expanded our approach to mixed analogue-digital circuit design and optimisation. Note that we restrict our research to the static technique of fitness function evaluation only.

3 Overview of Mixed Analogue-Digital Circuit Design and Optimisation Based on Evolutionary Probabilistic Algorithms

The design strategy used in our design flow is a performance-driven concurrent top-down down-up methodology. To tract complexity, a large design is broken up into a set of subblocks, until all blocks are at transistor level. These low-level blocks are

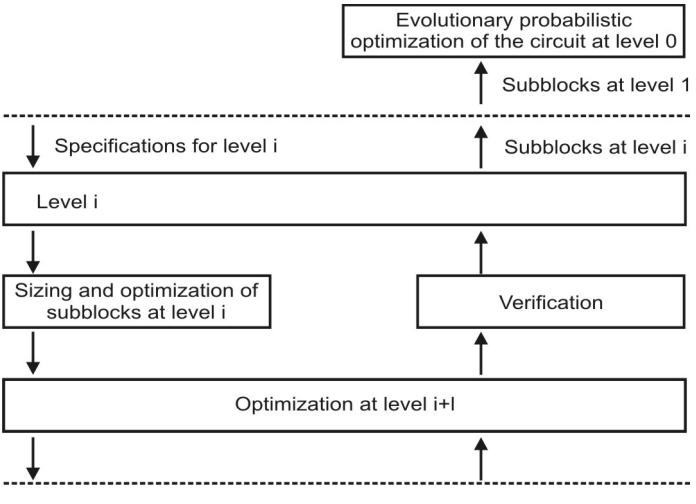


Fig. 1. A concurrent design flow implemented in our system

sized to be optimal. Then performance of the complete systems is checked. If it does not meet all specifications then optimisation at the system level and at the transistor level is done simultaneously. In the final optimisation loop the design space includes only crucial circuit parameters (for example, the transistors' widths and lengths, values of independent voltages sources, etc.). They are given by a user.

Fig. 1 illustrates the design strategy implemented in our system. Optimisation at low hierarchical levels can be performed either by a commercial tool or by evolutionary probabilistic algorithms. The final optimisation is performed by evolutionary probabilistic algorithms.

We use the linear circuit representation according to recommendations [15]. A genotype is formed by means of the combination of separate genes for each variable component (a transistor, an independent voltage source, etc.). Fig. 2 shows the general structure of our circuit representation.

UMDA [13] is an evolutionary algorithm that combines mutation and recombination by using probabilistic distribution. The distribution is estimated from a set of selected points *M*. Our focus is on the truncation selection, where $\tau = M/N$ is the amount of selected individuals, *N* is the population size. Selected individuals are then used to generate *N* new points for the next generation according to the probability *p(X, t)* in the population at the generation *t*

$$p(X,t) = \prod_{i=1}^{n} p_i^S(x_i,t) \tag{1}$$

where $p_i^S(x_i,t)$ are marginal frequencies;

n is equal to the genotype length; *n* is calculated as follows:

$$n = \sum_{i=1}^{N_p} (Max_i - Min_i)/D_i \tag{2}$$

where Max_i, Min_i, and D_i are equal to the maximal, minimal acceptable value and the incremental value of parameter *i* correspondently;

N_p is a number of varying parameters.

We use mutation settings according to recommendations [14].

The decision as to how many iterations should be done and evaluated can be based upon several factors such as termination conditions of an optimisation algorithm used, certain metric factor etc.

…	Transistor i		…	Independent voltage source j	…
…	Width	Length	…	Value	…
…	101..	01..	…	1110…	…

Fig. 2. A circuit representation genotype

4 Experimental Setup

The proposed evolutionary probabilistic approach has been prototyped in a software framework EvolCircuit. We have integrated in one system the existing in-house

simulation tool TITAN [16] and a new tool for probabilistic evolutionary optimisation. The system supports a combination of standard cells and custom cells. Standard cells have been optimised by WiCkeD tool and proved in silicon. Therefore, we do not change their parameters. Our focus is only on variations of crucial circuit parameters (transistors widths and length, independent voltage sources values, etc.).

Our circuit design benchmark chosen is a symmetry recognition circuit [2]. It contains analogue-digital converters, comparators, flip-flops, etc. Fig. 3 illustrates some key features of symmetry recognition.

The circuit has been designed by a human designer. However, design specifications did not meet. We have applied our approach to reach design goals. The input specification for the symmetry recognition circuit is summarized in Table 1.

For the evaluation of the fitness functions the values of the standard deviations of symmetry recognition are calculated across evaluation points *Nc*

$$F = \max\{\sqrt{\frac{1}{N_C}\sum_{i=1}^{N_C}(f_i - \bar{f}_{0.8})^2}\ |_{\frac{T_1}{T_2}\equiv 0.8}; \sqrt{\frac{1}{N_C}\sum_{i=1}^{N_C}(f_i - \bar{f}_{0.9})^2}\ |_{\frac{T_1}{T_2}\equiv 0.9}\} \quad (3)$$

$$\bar{f}_k = \frac{1}{N_C}\sum f_i\ |_{\frac{T_1}{T_2}=k}, k = 0.8; 0.9 \quad (4)$$

where f_i is the recognition asymmetry.

We assume that asymmetry has been recognized after the output voltage of symmetry recognition circuit [2] is below 1.0 V. Individuals with minimal fitness function are selected as best ones. The stopping criterion is a finding a circuit with fitness function below 0.0001.

Out target technology is the Infineon BiCMOS technology B6CA. Therefore, we restrict our design space according to the technology constraints (minimal transistors widths, etc.) and sizing rules [17] in order to avoid incorrect solutions. We use the correspondent B6CA transistors models. However, EvolCircuit can be used for different technologies. The correspondent service has been included in EvolCircuit.

Design heuristics are given as follows:

- The lengths of all transistors are equal to *l*.
- The width of the transistor of the current source should be set to w/V_g ,

  where *w* is the width of the transistors of P-channel current mirror (CM). In our experiments we set V_g to 2.1.

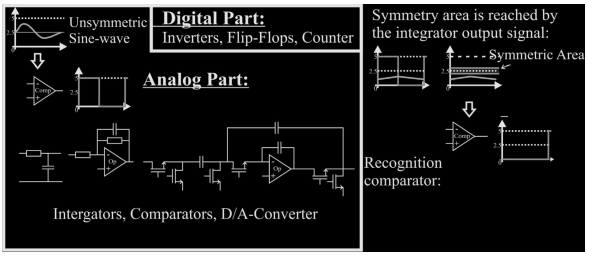


Fig. 3. Load-symmetry recognition circuit principle of a sine-wave

Table 1. Input specifications for our symmetry recognition design benchmark

Circuit Specifications	Values
Evaluation points	25 kHz, 125 kHz, 250 kHz, 500 kHz
Time asymmetry coefficient T_1/T_2	0.8; 0.9
Variations of DC voltage source	0.8 -2 V; Δ 0.1 V
Transistor length	1.5 -9.9 μm; Δ 0.3 μm
Transistor width of P-channel CM	5.3 -18.1 μm; Δ 0.1 μm
Par_2	1.0 -2.0; Δ 0.02

- The width of the transistors of inverters should be set to $w/2$.
- The width of the transistors of N-channel current mirror should be set to

$$w_N = \begin{cases} Par_2 * w, & w_N < 20 \quad \mu m \\ 20 \quad \mu m, & w_N \geq 20 \quad \mu m \end{cases}, \tag{5}$$

where w_N is the transistor width of N-channel current mirror.

5 Experimental Results

We examined the behaviour of different algorithms for the fixed number of bits n=22 with a truncation threshold τ from 0.2 to 0.5. Population size changes from N=4 to N=10. In despite of large design space dimension (more 4 millions possible solutions) we use small population sizes to reduce computational costs at system level. Figs. 4, 5 show how the best (curve 1) and average (curve 2) fitness functions are changed when algorithms settings are modified. The results have been obtained under assumption that the amount of iterations is less than the genotype length. Figs. 4, a, 5, a illustrate the case in which the population size varies between N=4 and N=10 and the truncation threshold is equal to τ=0.5. Figure 4, b shows the case in which the population size is fixed N=10, whilst the truncation threshold is decreased to τ=0.2.

The most obvious fact is that the search speed is generally enhanced where larger population sizes are used. However, computational costs are increased as well.

Therefore, we should keep a population size as small as possible to reduce the time to market. Truncation threshold varying enhances search capabilities in this

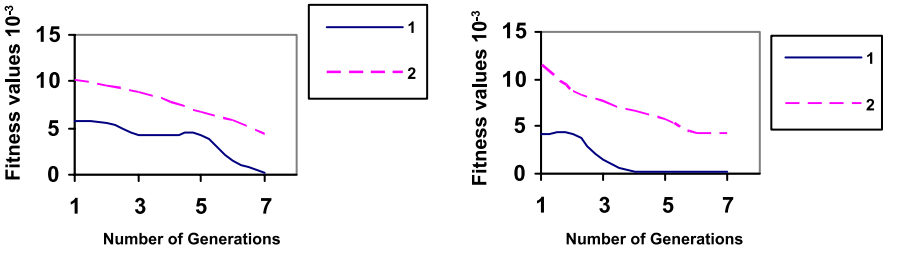


Fig. 4. Performance (average of 10 runs) behaviour of classical UMDA a) for population size N=4 and truncation selection with τ=0.5 b) for population size N=10 and truncation selection with τ=0.2

application insignificantly. In order to enhance design capabilities we have introduced elitism in evolutionary process. Figure 5, b illustrates the case in which the population size is equal to N=4 and the truncation threshold is equal to τ=0.5. It is obvious that elitism allows to keep a small population size and to decrease computational costs.

During evolutionary runs the required design specifications have been met. The optimised circuit has the required standard deviations of symmetry recognition. Fig. 6 shows the frequency performance of symmetry recognition of the optimised circuit (EC) and the human designed circuit (HD) for two asymmetry factors: 0.8 and 0.9.

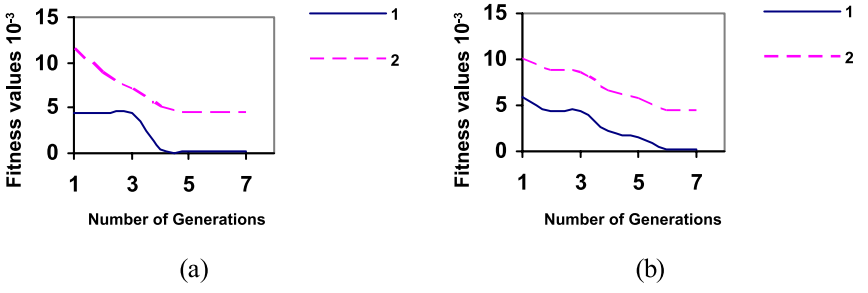


Fig. 5. Performance (average of 10 runs) behaviour a) of classical UMDA for population size N=10 and truncation selection with τ=0.5 b) of UMDA with elitism for population size N=4 and truncation selection with τ=0.5

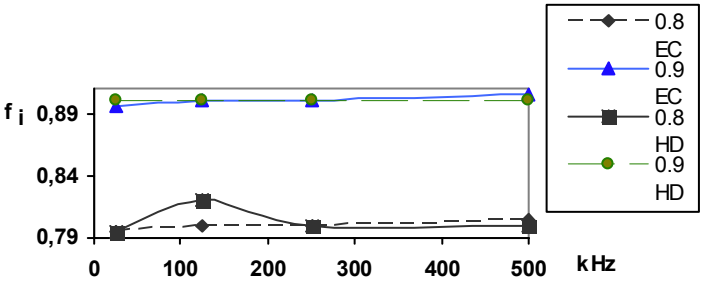


Fig. 6. Frequency performance behaviour of symmetry recognition circuit

6 Conclusions

In this paper the application of evolutionary probabilistic algorithms to mixed analogue-digital signal circuits design and optimisation was presented. The objective was to apply smart algorithms, e.g. evolutionary probabilistic algorithms, to optimisation at the system level. It was shown that both classical UMDA and its modifications with elitism can be used to improve design performance. It seems that UMDA with elitism is more suitable for industrial applications. Advantages of this approach are the reduced design time, overcoming design problems and meeting design specifications. The approach was validated by optimising symmetry recognition circuits containing analogue-digital

converter, flip-flops, comparators, etc. Experimental results validate the methodology by comparing the performance of the optimised circuit with the human designed circuit.

In this paper the results of single objective optimisation have been discussed. However, industrial design practice expects high yield of an optimised solution and small chip area. In the future research we will expand our approach to multi-objective optimisation.

Acknowledgments. This research is supported by a Marie Curie International Fellowship within the 6th European Community Framework Programme (grant MIF1-CT-2005-007950).

References

1. Rogenmoser, R., Kaeslin, H., Blickle, T.: Stochastic methods for transistor size optimization of CMOS VLSI circuits. In: PPSN IV. Springer-Verlag (1996) 849 - 858.
2. Nittayarumphong, S.et al.: Dynamic behaviour of PI controlled Class-E Resonant Converter for Step-Down Applications Using Piezoelectric Transformers. In: Proc. EPE2005. (2005)
3. Gielen, G. G. E., Rutenbar, R. A.: Computer-Aided Design of Analog and Mixed Signal Integrated Circuits. In: Proc. of the IEEE **12** (2000) 1825-1852
4. Gielen, G. G. E., et al.: Analog circuit design optimization based on symbolic simulation and simulated annealing. IEEE J. Solid-State Circuits **3** (1990) 707–713
5. Krasnicki, M., Phelps, R., Rutenbar, R. A., Carley, L. R.: Maelstrom: Efficient Simulation-Based Synthesis for Custom Analog Cells. In: Proceeding ACM/IEEE DAC. (1999)
6. Phelps, R., Krasnicki, M., Rutenbar, R. A., Carley, L. R., Hellums, J. R.: ANACONDA: Simulation based synthesis of analog circuits via stochastic pattern search. IEEE Trans. Computer-Aided Design Integr. Circuits Syst. **6** (2000) 703–717
7. Ochotta, E. S. , Rutenbar, R. A., Carley, L. R.: Synthesis of high-performance analog circuits in ASTRX/OBLX. IEEE Trans. CAD Integr. Circuits Syst. **3** (1996) 273–294
8. Krasnicki, M.J., Phelps, R., Hellums, J.R., McClung, M., Rutenbar, R.A., Carley, L.R.: ASF: a practical simulation-based methodology for the synthesis of custom analog circuits. In: Proceedings of ICCAD. (2001) 350 – 357
9. Aplaydin, G., et al.: An Evolutionary Approach to Automatic Synthesis of High-Performance Analog Integrated Circuits. IEEE Trans. Evol. Comp. **3** (2003) 240 – 252
10. Antreich, K., et al.: WiCkeD: Analog Circuit Synthesis Incorporating Mismatch. In: CICC 2000. (2000)
11. Gielen, G., McConaghy, T., Eeckelaert, T.: Performance Space Modeling for Hierarchical Synthesis of Analog Integrated Circuits. In: Proc. DAC 2005. (2005) 881 – 886
12. Mühlenbein, H.: The Equation for Response to Selection and its Use for Prediction. Evolutionary Computation **5** (1998) 303-346
13. Muehlenbein H., Mahnig, T.: Evolutionary Computation and Wright's equation, Theoretical Computer Science **287** (2002) 145-165
14. Mühlenbein, H., Zinchenko, L.A., et al.: Effective Mutation Rate of Probabilistic Models for Evolutionary Analog Circuit Design. In: Proc. IEEE ICAIS 2002. (2002) 401-406
15. Zinchenko, L.A., et al.: A Comparison of Different Circuit Representations for Evolutionary Analog Circuit Design. In: Proc. ICES. (2003) 13-23
16. Feldmann, U., Wever, U., Zheng, Q., Schultz, R., Wriedt, H.: Algorithms for modern circuit simulation. Archiv für Elektronik und Übertragungstechnik **46** (1992)
17. Graeb, H., Zizala, S., etc.: The Sizing Rules Method for Analog Integrated Circuit Design. In: Proc. IEEE ICCAD 2001. (2001) 343–349

Interactive Texture Design Using IEC Framework

Tsuneo Kagawa, Yukihide Tamotsu, Hiroaki Nishino, and Kouichi Utsumiya

Oita University, Dannoharu 700, Oita, Japan
{kagawa,yukihide,hn,utsumiya}@csis.oita-u.ac.jp

Abstract. In this paper, we propose a method to support texture mapping for intuitive designing of 3D objects in the scene of virtual or real space. To fit a texture pattern to the target 3D object, users should consider not only physical constraints, such as resolutions and scales, but also psychological constraints, such as fitness or moods. This method generates multiple candidate models applying various kinds of texture patterns and allows users to evaluate them sensitively. The technique called Interactive Evolutionary Computation (IEC) helps them to find the pleasant and adequate a texture pattern for the scene easily. Texture pattern is improved corresponding to users' evaluations. This framework provides a powerful environment for interactive texture design.

1 Introduction

Recently, 3D computer graphics (3D-CG) and virtual reality (VR) have become very popular and for many users because of rapid development of computer technology. 3D-CG and VR techniques are very important in many applications such as computer-aided design (CAD), games, films, and so on. Augmented reality (AR) and mixed reality (MR) also make progress with current advanced computer technologies. These technologies can provide enhanced capability of human-machine interaction to mixture view of virtual space and real world. When achieving an acceptable realistic AR or MR environment, especially when creating 3D-CG objects comfortable in the scene, texture mapping techniques play a very important role. Texture mapping can enhance visual richness on the surface of objects to construct realistic and high-quality 3D computer graphics and virtual space.

Texture patterns can be classified into procedural textures and photo-realistic textures. We utilize later one because we can construct texture patterns real and a broad range of patterns are prepared in advance. In this paper, we focus on how to synthesize 3D-CG objects in some "scene" of virtual space or real scene with photo-realistic texture patterns. To design a surface of 3D-CG object by some texture pattern captured by a digital camera or video to fit in the virtual or real environment, there are some constraints to fit the target objects in the scene as below:

1. **Scale:** The scale of texture patterns should be good for the scale of target 3D objects. This problem depends on the size of objects and contents in texture images. For example, if a brick pattern of wall is applied to the

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 439–448, 2007.
© Springer-Verlag Berlin Heidelberg 2007

small target object such as a small box, the box appears unnatural and has uncomfortable expression.
2. **Resolution:** When resolution of pattern is not large enough for the target objects, a texture pattern becomes dim and it may spoil the reality of the objects. Also when the ratio of length to breadth is not adequate for objects, the result becomes worse because it does not seem real.
3. **Fitness:** Selecting a texture pattern should correspond to a practical or a physics characteristic of objects. It depends of users' sensitivity. If a brick pattern is applied to the small target object such as a coffee cup, it must be strange but it heavily depends on their affinity or scene to build.
4. **Mood:** When the mood of the scene is a special one, texture pattern should fit to the atmosphere of the whole scene. For example, when the scene creates greasy mood, if the target object has very bright and beautiful surface, users may have uncomfortable feeling. So users must select an adequate pattern with users' sensitivity when applying texture images.

These constraints may not be defined explicitly and they heavily depend on users' intuitive sensitivity. To satisfy these constraints, some kinds of effects are added to the image and original images should be improved. So some improvement for the image is necessary to fit to an adequate texture pattern for a target 3D object. In many cases of improving texture patterns, users apply various kinds of image processing with a photo retouch or painting tools.

There are problems for editing texture patterns. As mentioned above, designing texture patterns adequate to the scene is a very sensitive work because users must take other objects in the scene into account. Since there is no definite rule to determine a favorite texture pattern design, it depends on the subjectivity of users. Furthermore, mapping 2D texture image onto 3D objects requires parameterization. It can reduce polygon constructing 3D objects to draw the surface in detail. And editing original image to add some effects cannot always add the same effects to 3D object. Users must take all these operations into considerattion when they create a new 3D objects. Therefore, texture synthesis is very difficult with trial and error process.

In this paper, we propose an interactive image processing method to find texture patterns adequate for 3D objects. The framework of interactive evolutionary computation (IEC)[1] can allow users to easily find more favorite texture patterns. Interactive genetic algorithm (IGA) is adopted for this method. Many works report that IGA can work well for the designing task such as 3D object creating or color scheme design. In this method, the system generates possible texture patterns with combination of image processing as candidates and users evaluate them. The most pleasant candidate can be obtained with repeating these steps.

2 Related Work

The texture mapping for 3D object is very simple but there are various combinations of parameters to acquire the pleasant texture pattern. Pure texture

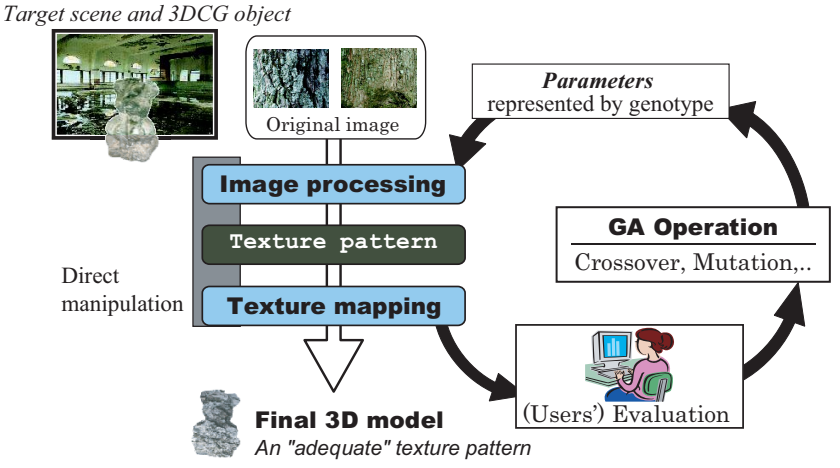


Fig. 1. An overview of our texture design support tool

synthesis has been big problem for a long time[6]. Many works about texture mapping method are proposed from various points of view[2,3,4,5] and they are successful to construct excellent 3D-CG. Many types of interactive texture synthesis are reported[8,9,10]. Our work attempts to adopt the framework of IEC. So users can find the most pleasant and adequate pattern without any special knowledge about texture synthesis or image processing.

As is well known, IEC based on GA can work effectively for such a searching problem[1,11,12]. This framework can provide a powerful function to support users to create a desirable 3D object with the most pleasant texture pattern. Since the impression of texture patterns largely depends on users' sensibilities, they are not able to easily find the most pleasant texture pattern[13,14,15,16].

3 Overview of IEC-Based Texture Synthesis

Our proposing method can support generating and resampling texture patterns from original images. Figure 1 shows a workflow of our supporting method. After repeating generation - evaluation loop, users can find the most comfortable and pleasant pattern for a texture of the target objects.

In this method as shown in figure 1, after users select some texture patterns arbitrary, a number of image processing, such as geometric transformation or color conversion, are applied to add effects to the selected texture image. After that, users can evaluate each candidate by adding scores and they are reflected to the candidates in next generation with IGA.

However, in the case that users can find and decide some kinds of effects for the texture pattern, this method allows direct manipulation for image processing. For example, in the dark scene, users may feel bright texture is not pleasant. It is required to operate image processing directly and not to operate corresponding

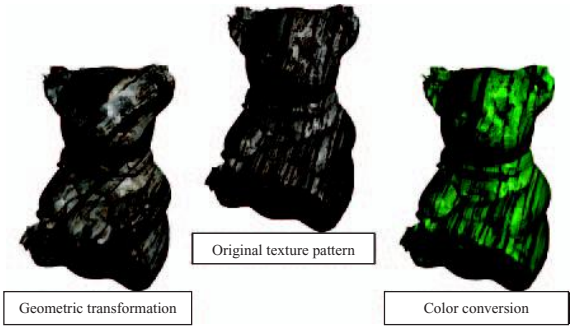


Fig. 2. Examples of effect by geometric transformation (Rotation θ=45), and Color conversion

part of genotype code. Currently they are "clamped", so the codes are fixed while evaluating process.

Image processing is classified into three types in our method, texture pattern extraction, geometric transformation and color conversion. Currently, these are essential process in the field of image processing implemented on photo-retouch tools. Figure 2 shows examples of implemented image process for adding some effects. Genotypes are coded to represent groups of operations and parameters of these processing. Parameters represent parameters, such as threshold, values and number of times. These processing must be independent mutually not to be influenced by order of them.

4 Image Processing

4.1 Image Selection and Extraction of Texture Patterns

Original images are stored and classified into several categories, such as regular / random, or rough / smooth, by users beforehand. These categories and similarities between each image are defined by the users to utilize for coding genotypes in GA. Constructing the texture databases automatically is our future work. The resolution of images is basically 1600 x 1200 pixels, which is large enough to map onto the 3D-CG objects. First, for every candidate, an image is selected and a texture pattern is extracted from this image. Next, parameters, such as center points in the texture pattern, height and width are calculated. These parameters are shown in figure 3(a).

4.2 Geometric Transformation

Geometric transformation of 2D texture image can affect appearances of the target objects. For example, if texture patterns are vertical (i.e. wooden grain), it looks like stable. Geometrical transform may fit the pattern to the object which

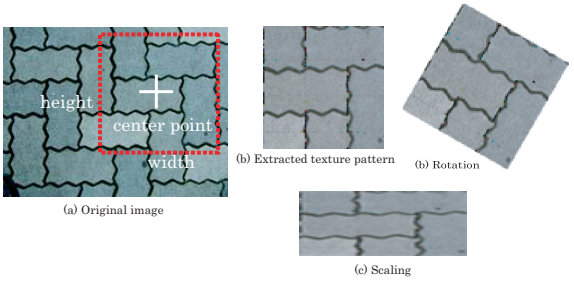


Fig. 3. Geometric transformation with affine transform. (a) Texture pattern extraction from original image. However, height and width are constrained to 2^m. (b) is extracted texture pattern, (c) is rotated by rotation matrices with $p{=}\pi/4$. (d) is scaled pattern, a is 1.2 (height), d is 0.4(width).

form is not simple. However, it is not perfect to apply these transformed textures. So we demand to consider texture morphing or warping method without distortion, as described in [4]. In this paper, we adopt simple 2D affine transformation that is independent of the feature of 3D model.

A selected image is transformed geometrically, such as distortion, geometric contraction, expansion, dilation, reflection, rotation, shear, similarity transformations, and spiral similarities. These are registered as geometric transformation with an affine transformation, which can preserve co-linearity and ratios of distances after any transformation. In general, an affine transformation is a composition of rotations, translations, dilations, and shears. So geometric transformation is described as below:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} \cos p & \sin p \\ -\sin p & \cos p \end{pmatrix} \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} e \\ f \end{pmatrix} \tag{1}$$

a and d are scaling parameter, a is for height, d for width. b and c are for shears, e and f are translation parameters. p represent an angle for rotation. Figure 3(b),(c),(d) shows examples of effects by geometric transformation.

4.3 Color Conversion

Transforming color also affects the impression of the 3D object appearance. Color conversions transform all colors in the texture pattern without varying relationship between each color. This process, first, projects all RGB colors into HIS (Hue, Intensity and Saturation) color space as shown in figure 4. It represents color distribution of a texture pattern in HIS color system. Next, arbitrary rotation of this color cluster generates color transformation with maintaining intensity and saturation, which affect texture expressions. An example for color conversion is shown in figure 5.

This process includes sharpening or blurring. Some spatial filters for image processing are adopted for sharpening, smoothing and blurring. Laplacian filter

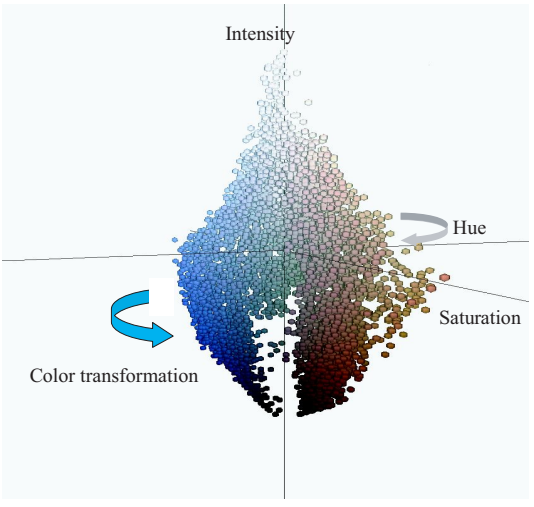


Fig. 4. Color distribution in a texture image. Color transformation is achieved with rotation in HIS color system around intensity axis.

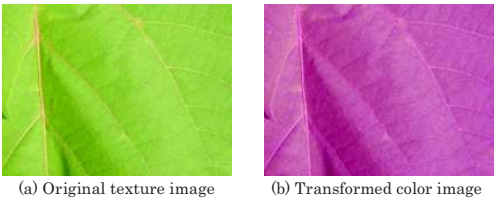

(a) Original texture image (b) Transformed color image

Fig. 5. An example of color transformation in a texture image. (a) is original texture image, (b)Only hue is transformed.

is utilized for sharpening, and Gaussian smoothing filter is for blurring shown in figure 6.

4.4 Lighting and Material Properties

Lighting works very effectively when rendering 3D objects. However, assume that the user can select these parameters such as lighting position, brightness, and color, the appearance heavily changes and it may cause unacceptable divergence. And the light position of the background scene must be considered.

To avoid these problems, we arrange 2 or 3 types of lighting configuration including “no light” in advance and give a code for each arrangement. The interface also provides function to facilitate and preset lighting parameters easily and interactively. Another research [13,15] describes about lighting arrangement with IEC.

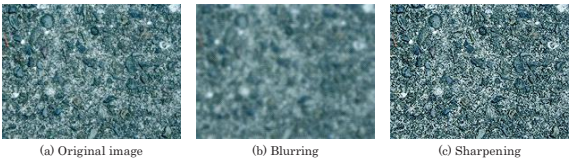


Fig. 6. Spatial filter application. (a) Original image, (b) Gaussian filter is applied for blurring, (c) Laplacian filter is applied for sharpening.

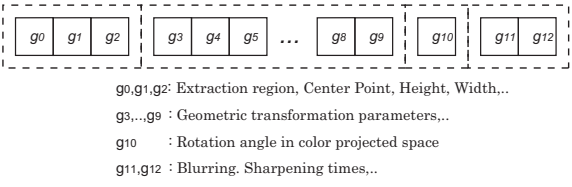


Fig. 7. Genotype coding. Currently, number of genotype is 12, but it changes according users' requirement. Furthermore, every parameter consists of several bits, so the length of genotype is longer than 12.

5 Interactive Evolutionary Computation

5.1 Genetic Algorithm

Figure 7 shows a genotype. Every parameter consists of several bits. The genotype mainly consists of four parts. These parts of codes express the parameters for each process respectively as follows:

1. **Image selection and extraction parameters:** Geometric parameters define texture pattern extraction from an original texture image. These parameters are 1.) coordination of center position, 2.) height and width of an extracting texture pattern. In the future, this part includes an ID number of image in the database.
2. **Affine transform:** These parameters represent the elements of affine transform matrix. All parameters are a, b, c, d, e, f, and p elements of affine transform matrix as shown in 4.2.
3. **Rotation angle for the color distribution in the HIS color system:** It is only one rotation angle parameter for color conversion described in 4.3. Color space rotates around intensity-axis.
4. **Spatial filter application:** These parameters represent the number of times to apply each filter for the texture pattern. These are the number of times to apply blurring and sharpening.

Currently, a simple GA is utilized for evaluation phase in this method. 9 chromosomes are utilized. Roulette wheel method is applied for genotype selection.

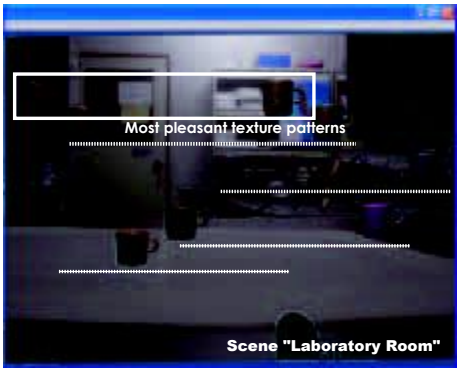


Fig. 8. Interface. Candidates are set in the target scene.Users can evaluate by changing the positions of candidates.

Multi-point crossover occurs by 75% and mutation occurs by 5%. When new chromosomes are reproduced in the new generation, the method is processed according to each chromosome. Users can evaluate these candidates as 1(Not Good) to 5(Good).

5.2 User Interface

We implemented the proposed method as a prototype system conducted preliminary evaluation as shown in figure 8. All the candidates are displayed in a virtual or real target scene for users to easily find adequate texture pattern in the scene. They can move the candidates in the scene so as to try various situations. In evaluation phase, Users put the candidates to locate score steps corresponding evaluation score. If users locate pleasant candidate on the highest place on the steps, the candidate is scored as maximum 5 points as shown in figure 8.

6 Experimental Results

8 persons evaluate our tool to overview the effectiveness of our method. They have knowledge for creating 3D-CG contents and texture synthesis. We preset their tasks to construct a vase in a certain scene in Japanese room shown in figure 9. An average of repetition times (scoring and evaluation) is 5.0 times, and it takes about 65 seconds. We also interviewed them about valuation of our tool. First, as they score for it 1(Not good) - 5(Good), the average of these valuation score is 3.5. This seems good value (but not very good).

After evaluation test, there are some comments to require the function to change texture pattern manually. However, they can recognize that this method enable users to search an adequate texture pattern without any special knowledge for texture syntethis. More consideration is required to this point. We must



Fig. 9. Experimental interface. Users find the texture pattern for vase in the Japanese room.

compare our method to normal method for texture mapping and analize effectiveness of adopting IEC framework. There is other comment to display more various texture patterns. Our tool tends to vary colors more than geometries. So it seems only colors are changing without geometric transformation.

7 Conclusions

In this paper, we describe about an interactive texture synthesis method based on the IEC. This tool provides the interactive texture synthesis in some 3D or 2D scene with combining various types of image processing. IEC framework supports to explore the optimum and suitable texture pattern sensitively and 3D scene representation helps to achieve simple texture mapping.

Simple texture synthesis method enables users to investigate adequate image processing for texture pattern construction fitting to a scene intuitively and easily. Although combination of transformation of the 2D texture patterns can generate unexpected new texture pattern, users will find a new type of texture pattern. This method can be applied for the design of manufacturing or artifact products. But an exhaustive mapping method is required to apply the texture obtained by this method.

However, the current experiment is not large enough for evaluating our method. We must add test users to obtain more objective results and analize them more in detail.

Our future work is first to enlarge variety of image processing. Next, we should consider how to intelligibly display the results 3D objects. Additionally, it is necessary to use more detailed information about texture image and varying them in the evolutional process. The proposed technique should be evaluated in an actual design process.

References

1. Takagi, H.: Interactive Evolutionary Computation: Fusions of The Capabilities of EC Optimization and Human Evaluation. Proc. of the IEEE, vol.89, No.9, pp.1275–1296, 2001.
2. Debevec, P: Image-Based Modeling and Lighting. Computer Graphics, Nov. pp.46–50, 1999.
3. Jia, J., and Tang, C.,: Inference of Segmented Color and Texture Description by Tensor Voting. Trans. of PAMI, Vol. 26, No.6,pp.771–786, 2004.
4. Beauchesne, E., and Roy., S.: Automatic Relighting of Overlapping Textures of a 3D Model. Proc. of CVPR 2003, pp.1-8, 2003.
5. Isenburg, M., and Soneylink, J.: Compressing Texture Coordinates with Selective Linear Predictions. Proc. of CGI 2003, pp.126-131, 1999.
6. Cha Zhang, and Tsuhan Chen: A survey on image-based rendering–representation, sampling and compression. Signal Processing: Image Communication, Vol.19, pp.1–28, 2004.
7. Matthias Zwicker, Mark Paulym, Oliver Knoll, Marcus Gross, and ETH Zürich: Pointshop 3D: An Interactive System for point-Based Surface Editing. Proc. SIGGRAPH '02, pp.322–329, 2002.
8. Steve Zelinka, and Michael Garland: Interative Texture Synthesis on Surface Using Jump Maps. Proc. Eurographics Symposium on Rendering 2003, pp.90–96, 2003.
9. Jie Zheng, Hongbin Ji, and Wanhai Yang: Interactive PC Texture-Based Volume Rendering for Large Database. Proc. IEEE ICICIC'06, pp.350-353, 2006.
10. Patricio Parada, Javier Ruiz-del-solar, Wladimir Plagges, and Mario Köppen: Interactive Texture Synthesis. Proc. of the 11th Int. Conf. on Image Analysis and Processing, ICIAP, pp.434-439, 2001.
11. Aoki, K., and Takgi, H.: 3D CG lighting with an Interactive GA. Knowledge-Based Intelligent Electronic Systems, KES '97, vol.1, pp.296-301, 1997.
12. HS, Kim, and SB, Cho: Application of Interactive Genetic Algorithm to fashion design. Engineering Applications of Artificial Intelligence, vol.13, No.6, pp.635-644, 2000.
13. Nishino, H., Hieda, M., Kagawa, T., Takagi, H., and Utsumiya, K.: An IEC-based 3D Geometric Morphing System. Proc. of the IEEE SMC2003, pp.987–992, 2003.
14. Kagawa, T., Nishino, H., and Utsumiya, T.: A Color Design Assistant based on User's Sensitivity. Proc. of the IEEE SMC2003, pp.974–979, 2003.
15. Nishino, H., Aoki, K., Takagi, H., Kagawa, T., and Utsumiya, K.: A Synthesized 3DCG Contents Generator Using IEC Framework. Proc. of the IEEE SMC 2004, pp.5719-5724, 2004.
16. Kagawa, T. Gyohten, K., Nishino, H., and Utsumiya, K.: A Creative Color Design Support Tool Based on A Color Reflection Scheme. Proc. the 8th International Conference on VSMM2002, pp.372–378, 2002.

Towards an Interactive, Generative Design System: Integrating a 'Build and Evolve' Approach with Machine Learning for Complex Freeform Design

Azahar T. Machwe and Ian C. Parmee

ACDDM Lab, Faculty of CEMS, University of the West of England, Frenchay Campus, Bristol, United Kingdom, BS16 1QY
{azahar.machwe,ian.parmee}@uwe.ac.uk

Abstract. The research presented in this paper deals concerns interactive evolutionary design systems and specifically with the Interactive Evolutionary Design Environment (IEDE) developed by the authors. We describe the IEDE concentrating upon the three major components: Component-based Representation; Construction and Repair Agents (providing build and evolve services) and a machine learning sub-system. We also describe the clustering technique utilized within the IEDE to improve the user interactivity of the system.

Keywords: Machine Learning, User-centered Evolutionary Design, Design Representation, Agency.

1 Introduction

Conceptual design is the first (and the most critical) stage of any design process. In this stage of the design process design activity includes a substantial subjective element due to a lack of well-defined models to allow machine based testing of the generated concepts. While it may be possible to machine-test certain features of a generated solution, a full analysis can only be performed by the designer using experiential knowledge and domain expertise. A designer-based evaluation is critical when the aesthetics of the object being designed are very important. Even though certain global aesthetic rules exist they cannot fully evaluate the aesthetic appeal of a design due to inherent highly subjective aspects. Designer-centered evolutionary systems therefore involve various degrees of human interaction within the evolutionary design process which extends from having a purely human based evaluation of solution to a mixture of machine / human based evaluation [1, 2, 3, 4].

This paper highlights the various novel features of the Interactive Evolutionary Design Environment (IEDE) developed by the authors for the interactive design of urban furniture [5]. The current work is based on the interactive evolutionary Bridge design system [6] which dealt with the design of simple 2-D simply supported span bridges (see Figure 1).

The current work builds upon the Bridge design system by extending the design into 3-D space as well as increasing the subjective element within the solutions. As

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 449–458, 2007.
© Springer-Verlag Berlin Heidelberg 2007

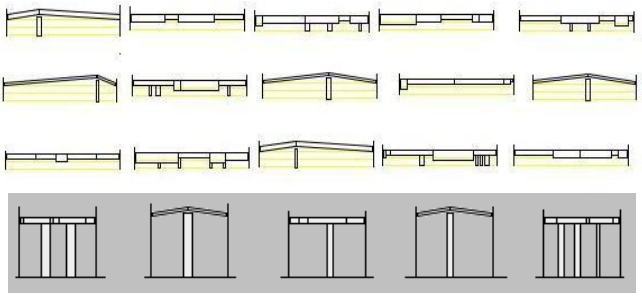


Fig. 1. Some non-optimized (*top*) and optimized (*bottom*) solutions from the Bridge Design System

with the Bridge design system, the solutions within the current system are evaluated by a machine-based fitness function as well as by the designer. Since continuous evaluations by the designer can lead to fatigue and cognitive overload resulting in poor evaluations, the IEDE includes a machine based learning system to assimilate the preferences of the designer during run-time. The IEDE also uses a variable length component based representation for the designs which allows for a wide variety of solutions to be presented to the designer [5]. This essential feature allows the conceptual design system to present the designer with a wide variety of concepts rather than prematurely narrowing down design options.

1.1 Major Components of the IEDE

The major components of the IEDE are shown in Figure 2. The IEDE uses a unique Construction Agent (C.A.) [5] approach to assemble the initial solutions based on a set of rules. The Repair Agent (R.A.) is mainly responsible for ensuring that solutions do not violate any constraints. Using the Agent paradigm we can incorporate different behaviors within the Construction and Repair Agents (C.A.R.A.) which allows for the assembly of a wide variety of solutions ranging from totally freeform (components placed randomly in space) to rigid well defined (components placed according to a layout plan). Further details are given in section 4 of this paper.

Another important component of the IEDE is the machine learning sub-system. This is a Case Based Reasoning system (CBR) which learns the preferences of the designer at run-time. This is described in the section 5 of this paper.

An evolutionary search and exploration engine lies at the heart of the system. Since most of the evolutionary operations (such as Mutation) have been delegated to the C.A.R.A. the engine can use any evolutionary algorithm. This flexibility also allows us to implement more than one search technique within the engine as can be seen in Figure 2 where we have incorporated a secondary Local Search Agent. Currently a simple Evolutionary Programming algorithm is being used as the primary search and exploration algorithm.

In this paper we will concentrate upon the representation (including Clustering), C.A.R.A. and the machine learning sub-system. For a detailed overview of the various components the reader is directed to [5].

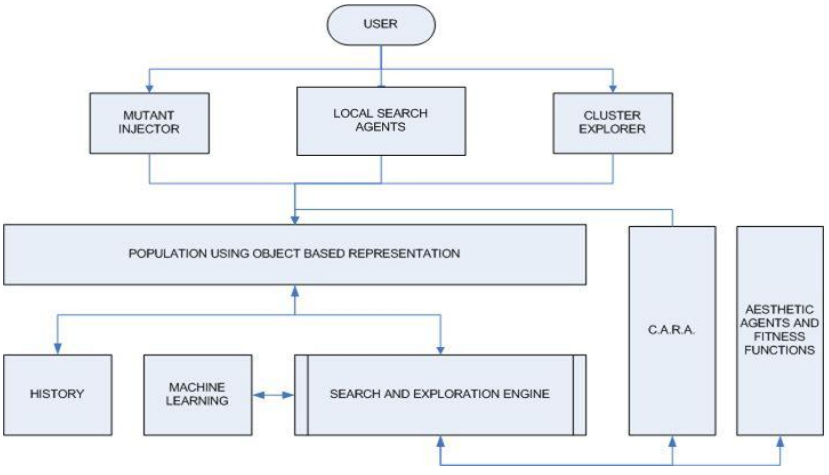


Fig. 2. Major components of the IEDE.

2 Representation

It is widely known that a fixed length parametric chromosome cannot provide the same level of exploration as a variable length one [2]. As Bentley and Corne [4] suggest exploration truly takes place only when variable parameters define a set of components from which the solution is constructed as opposed to directly defining the solution. Having a fixed-length parametric representation of a problem only allows the evolutionary algorithm (EA) to search within the space defined by the fixed-length of the chromosome. In other words a component based representation can 'vary the dimensionality of the space [being explored] by adding or removing elements' [4]. Furthermore during the initial stages of a design process solution exploration is a primary requirement. A variable length object (i.e. component) based [7] representation is used within the IEDE providing a robust and flexible platform for evolutionary search and exploration. [5]

The major disadvantage of a component based representation is that the solutions are dependant upon the manner in which the design divided into components. The urban furniture, in the case of the simple bench, comprises three types of components namely: Seat, Leg and Back. There are various other ways of decomposing a bench design into components (such as having Supporting and Supported Elements) but using a rule-based construction and repair agent allows us to offset this disadvantage to a certain extent and enhances the flexibility of the component based representation.

As can be seen from Figure 3 and 4 each component type can contain a variable number of components (or Elements) and each Element has a set of properties such as style, position and dimensions. Looking at the example solution given in Figure 4 we see it contains 3 Seat Elements, 2 Leg Elements and 1 Back Element.

Since the IEDE is user-interactive the solutions must have a subjective aspect to them so as to allow user based evaluation. This is incorporated within the representation by using different element styles (see Figure 5). The overall fitness of

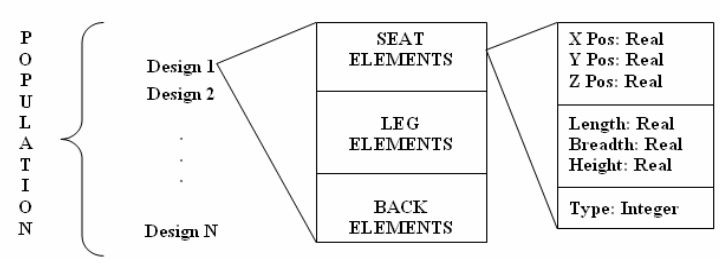


Fig. 3. Component based representation used within current IEDE

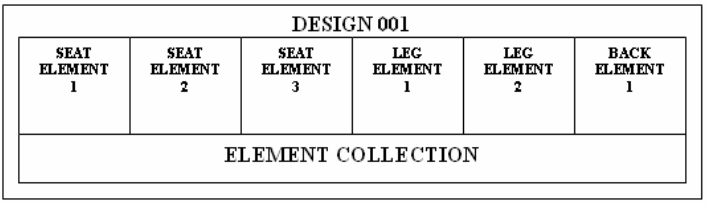


Fig. 4. An example of a Bench solution



Fig. 5. An example of different Element styles: Grill (*left*), Solid (*center*) and T-shaped (*right*)

the solution is a combination of subjective (user based) as well as objective (machine based) analysis of the solution, the configuration and *style* of component contributes equally towards the overall fitness of the solution since the machine based analysis (such as buckling analysis of supports) is modified based on which type of element is present. The style also affects certain properties of the Element such as weight and volume. The subjective (aesthetic) impact of styling with respect to user assessed fitness is also quite apparent. Another advantage of using a component based representation with styling of components is that the richness of the representation could be enhanced by adding new styles of components. Initially in the Bridge design system there were just two styles of components which increased to three in the initial implementation of the Bench design system and six in the current version.

3 Clustering and Local Search

As stated before variable length chromosomes allow for a wider exploration of the solution space by changing the dimensionality of the problem. Therefore within the

Bench design system we utilize clustering based on the number of different component types (style of component is ignored) present within a solution. Clustering is not used for the purpose of fitness assignment [8] instead here clustering enables us to improve the presentation of solutions to the user as well as to use a much larger population size than usually possible within Interactive Evolutionary Systems.

The clustering has been implemented in the following manner:

1) Parent Population at generation 't' (**P**(t)) is mutated to give Child Population (**C**(t)).
2) **P**(t) and **C**(t) are combined into one set **S**(t).
3) Clustering is performed on **S**(t) on the basis of number of different component types.
4) The user is shown the solution with highest fitness (machine based) from the cluster and has the option of exploring each of the clusters in detail.

Therefore if a population of size **N** is mutated and combined with the child population the total population becomes **2N**. Since the clustering is done based on number of component types which doesn't change upon mutation of a solution (in the current version) the number of clusters remains the same. This ensures that the number of clusters does not exceed a certain limit. Furthermore showing the user a single solution from a cluster containing both parent and child solutions ensures that maximum exploration has taken place before presenting the user with the results. Therefore by looking at the cluster representatives the user is *indirectly* evaluating **2N** solutions and has the *option* of *directly* evaluating any of the **2N** solutions by exploring the clusters.

Each combination of a number of Seat, Bench and Leg Elements (i.e. the component types) represents (in this system) a separate search space (for optimization) or concept, for example a solution having no Back Elements and normal Legs and Seat represents the concept of a 'Backless Seat'. User-based testing involving multiple users (variable length runs) showed that while working with such a representation the user usually performs a higher level space (or concept) selection first which is then followed by a search within the selected space (thus reducing to a traditional interactive evolutionary system). This can be seen from the fact that the number of clusters reduces rapidly with generations till there are two or three clusters (concepts) present (see Figure 6). The reduction in number of clusters is less rapid when only a few clusters remain and the user is involved. This is expected behavior since if we use a purely machine-based fitness function then the EA will encourage the growth of the cluster containing solutions of the fittest type in terms of quantitative criteria whereas when we include qualitative (user) assessment the user may not be able to decide quickly between two different solution types and would tend to favor both as much as possible.

Another advantage of using clustering is that as the generations progress the user is shown a lesser number of solutions for evaluation (the number of clusters reduce and the user is shown only the fittest solution from each cluster) but each solution shown is the fittest within a larger set of solutions.

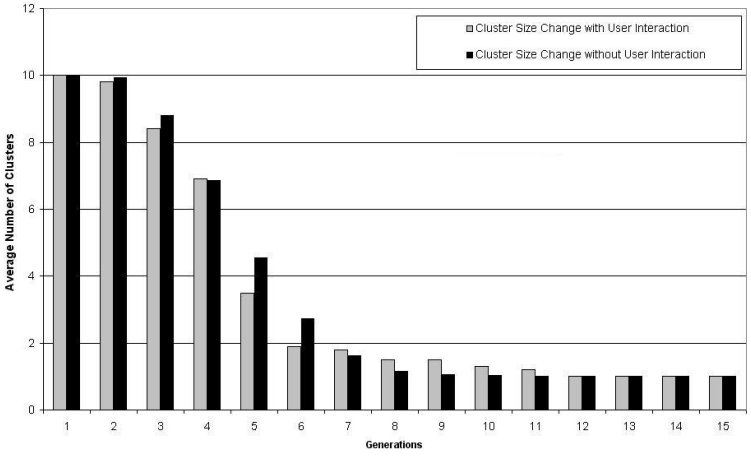


Fig. 6. Reduction in number of clusters as generations (*X-axis*) progress. Lighter bars show decrease with user interaction and darker bars show decrease with just the machine-based analysis of solutions.

3.1 Local Search Agent

While it is important to concentrate on exploration of variable dimension search space we cannot ignore fixed dimensional optimization. This is so because when the user is evaluating designs subjectively it would be useful to see how solutions would behave if they are optimized using the machine based fitness functions. Furthermore since a weighted sum [5] approach is being followed to compute the net fitness, the user may find it useful to see how the selected solutions behave with different weights given to the various fitness criterions. Keeping the above in mind a Local Search Agent has been integrated with the IEDE which uses a simple hill-climbing algorithm to search the user-selected search space. The starting solution is the solution selected by the user.

Another important reason for using a Local Search Agent is that once the number of clusters converges we move from variable dimensional search and exploration to fixed length optimization. Within the fixed length optimization space while the number of components remains the same their physical configuration can change. Therefore many times the solution really liked by the user may loose its aesthetic appeal once optimized.

4 Construction and Repair Agent (C.A.R.A.)

The Construction and Repair Agent consists of two components, specifically the Construction Agent (CA) and the Repair Agent (RA). The main task of the Construction Agent is to assemble the initial population using a set of rules and ensuring that the constructed solutions do not violate any constraints (such as extending outside the bounding space). The Repair Agent is responsible for ensuring that the solutions continue to remain within constraints throughout the evolutionary

process since exploratory operations (such as mutation) can disrupt the designs. Both the Agents work on a set of rules. The more flexible the rules the greater will be the variety of designs generated by the CA. Thus using this rule-based construct and repair approach allows us to obtain some of those advantages possessed by systems which don't evolve designs but evolve rules that can grow a design. For instance, providing high flexibility in what is actually represented since a flexible representation is of no use if the solutions that it is representing are highly rigid.

Figures 7a and 7b show the impact of the nature of the rule-base on the assembled and subsequently evolved solutions. When we use a flexible rule-base (Figure 7a) we see that the resulting solutions often have different configurations. When we use a rigid rule base (Figure 7b) we find that while there are structural changes between the assembled and evolved solutions, the overall configuration of the design remains the same.

5 Machine Learning

While Clustering and the Local Search Agent are quite useful for the user one of the central problems with user interaction remains the problem of user fatigue. The quality and efficiency of user evaluations degrades with the number of solutions evaluated. This is one of the primary reasons for using small population sizes which would converge rapidly especially when the solutions are being evaluated only by the user and there is no machine-based analysis component. There is a clear need to provide some kind of support to the designer. Limited support can be provided by transferring a part of the fitness analysis to the machine (such as by analyzing the aesthetic fitness using a set of rules [6]) and showing the user only the elite solutions from a generation. But even in this case the user may have to evaluate a large number of solutions.

The structure of the machine learning sub-system within the IEDE is given in Figure 8. It is a Case Based Reasoning (CBR) system where each user-evaluated design is encapsulated within a Case (similar to [12]) and then stored within the Case Library. The retrieval mechanism is based on design similarity (number of elements, shape, style etc.). The machine learning process is also aided by the convergence of the population. An advantage of using CBR is that the design information can be stored without modifications which may lead to loss of essential information [10]. It is also widely known that for conceptual design the CBR approach is quite promising [11]. Furthermore, testing with alternative methods such as Fuzzy Rule based and Radial Basis Functions with the Bridge design system did not show the desired performance required for online learning [12].

Testing of the machine-learning sub-system was carried out by eight different users with at least five runs per user. The average run length over multiple users and runs was approximately 5 generations. Population size of N=40 was used with a tournament size of three. The users were asked to check the machine supplied rank of the solutions shown and if they disagreed with the machine rank they could modify it. A record was kept of the number of modifications made to the machine supplied rank each generation as the runs progressed. The default machine supplied rank was 'zero'.

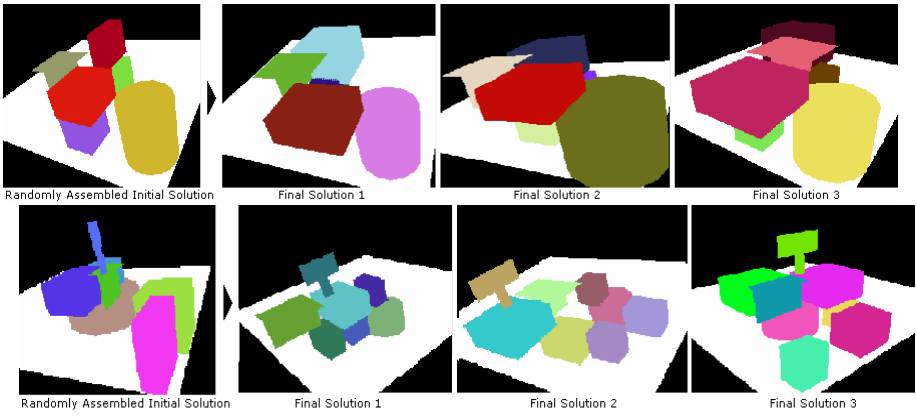


Fig. 7a. Initial solutions (*extreme left*) and three evolved results using a flexible rule base

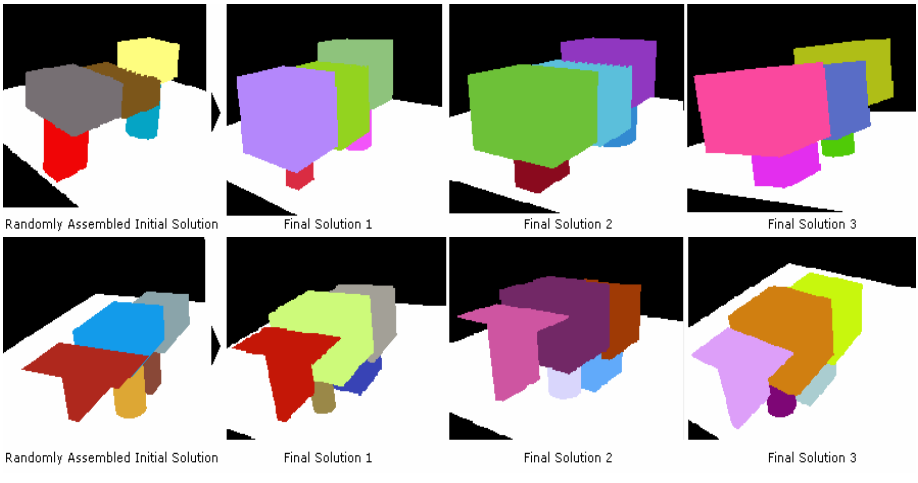


Fig. 7b. Initial solutions (*extreme left*) and three evolved solutions using a rigid rule based

The results are shown in Figure 9 where we can see the average (over all the users / runs) acceptance levels of machine supplied ranks increase with generation.

An interesting feature of the results is a slight decrease in the percentage of user's acceptance levels towards the middle. This may represent the user changing their mind about the fitness of a solution which leads to the rejection of machine supplied rank which was being agreed to in the previous generations. It can also represent a mid-course correction where the system was converging to alternative solutions (based on the objective analysis of the solutions) and the user is forced to increase the fitness levels to steer the path of evolution towards the aesthetically pleasing solutions.

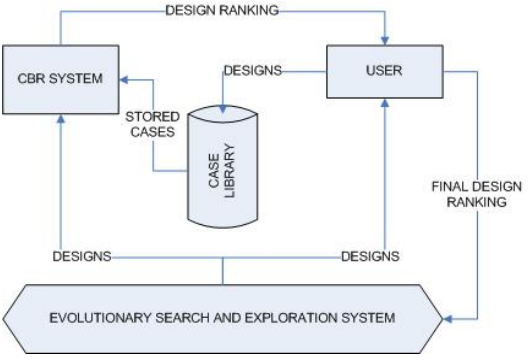


Fig. 8. The Case Based Reasoning System implemented within the Bench Design IEDE

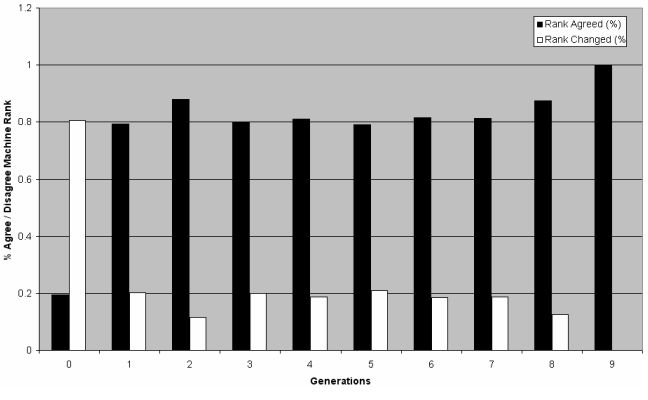


Fig. 9. Increasing user acceptance of machine supplied rank (Y-axis) with generations (*X-axis*). Average percentage of machine supplied solution ranks (in a generation) accepted by the user are given in black and average percentage of rejected ranks are given in white.

6 Conclusion

In this paper we have highlighted the major components of the urban furniture IEDE developed by the authors. We have also shown how the Representation, C.A.R.A. and Machine Learning components work together to create a design environment which allows both subjective assessment as well as machine based structural analysis of solutions.

We have shown how a machine learning system can reduce the load of continuous evaluations placed on the user in interactive evolutionary systems. When this is combined with the build and evolve approach being used we have all of the necessary components required for a generative system. This will learn from the user-evaluated solutions and feed information back to the Construction Agent rule base which is them modified to create similar solutions. This will ensure that the number of solutions shown to the user do not decrease with the convergence of the system. As

the number of clusters reduces the missing solutions can be replaced by the new solutions generated by the Construction Agent. This will also increase the competition for the user assigned fitness.

References

1. Gero, JS (2002) Computational models of creative designing based on situated cognition, in T Hewett and T Kavanagh (eds.), Creativity and Cognition 2002, New York, ACM Press, USA.
2. Bentley, P.J. ed. (1999). Evolutionary Design By Computers. 1st Edition. Morgan-Kaufmann, USA.
3. Parmee, I.C. (2002). Improving problem definition through interactive evolutionary computation. Artificial Intelligence for Engineering Design, Analysis and Manufacturing (2002), 16(3), pg. 185-202. Cambridge University Press, Printed in USA.
4. Bentley, P.J. and Corne, D.W. ed. (2002). Creative Evolutionary Systems. 1st Edition. Morgan-Kauffmann. USA.
5. Machwe, A. and Parmee, I.C. (2006) Integrating aesthetic criteria with evolutionary processes in complex, free-form design – an initial investigation. Congress on Evolutionary Computation – 2006, Vancouver, Canada.
6. Machwe, A. Parmee, I.C. and Miles, J.C. (2005) Integrating Aesthetic Criteria with a user-centric evolutionary system via a component based design representation. Proceedings of International Conference on Engineering Design (2005), Melbourne, Australia.
7. Machwe, A. Parmee, I.C. and Miles, J.C. Overcoming representation issues when including aesthetic criteria in evolutionary design. Proceedings of ASCE International Conference in Civil Engineering (2005) ,Mexico.
8. H.S. Kim and S.B. Cho. An efficient genetic algorithm with less fitness evaluation by clustering. pages 887{894. Proceedings of the 2001 IEEE Congress on Evolutionary Computation, Seoul, Korea, 2001.
9. I.C. Parmee. Evolutionary and Adaptive Computing in Engineering Design. Springer Verlag, 2001.
10. Kolodner, J. "Case-based Reasoning". Morgan Kaufmann Publishers 1993.
11. Mitchell, T.M. "Machine Learning". McGraw Hill International.1997.
12. Machwe, A. and Parmee, I.C. (2006) Introducing Machine Learning within an Interactive Evolutionary Design Environment. International Design Conference – Design 2006, Croatia.

An Interactive Graphics Rendering Optimizer Based on Immune Algorithm

Hiroaki Nishino, Takuya Sueyoshi, Tsuneo Kagawa, and Kouichi Utsumiya

Department of Computer Science and Intelligent Systems
Oita University, 700 Dannoharu, Oita 870-1192, Japan
{hn,takuya,kagawa,utsumiya}@csis.oita-u.ac.jp

Abstract. We propose an interactive computer graphics authoring method based on interactive evolutionary computation (IEC). Previous systems mainly employed genetic algorithm (GA) to explore an optimum set of 3D graphics parameters. The proposed method adopts a different computation model called immune algorithm (IA) to ease the creation of varied 3D models even if a user doesn't have any specific idea of final 3D products. Because artistic work like graphics design needs a process to diversify the user's imagery, a tool that can show the user a broad range of solutions is particularly important. IA enables to effectively explore a global optimum solution as well as other multiple quasi-optimum solutions in a huge search space by using its essential mechanisms such as antibody formation and self-regulating function.

1 Introduction

Recent advances in 3 dimensional computer graphics (3DCG) technology allow the public to install and enjoy real-time animation software on their personal computers. Even the creation of their own contents by using the 3DCG authoring tools becomes possible. However, there is a big hurdle to cross over before mastering the 3DCG authoring techniques such as the learning of 3DCG theories, becoming familiar with a specific authoring software tool, and honing an aesthetic sense to create a good 3D product.

We have been working on the development of some 3DCG authoring techniques which allow even novices to intuitively acquire contents production power without paying attention to any details on the 3DCG theories and authoring skills [1][2]. We have been applied a technical framework called interactive evolutionary computation (IEC) [3] to realize the intuitive graphics authoring environment. As shown in figure 1, a user simply looks at multiple graphics images produced by the system and rates each image based on his/her subjective preference. He/she gives his/her preferred images higher marks and vice versa. Then, the system creates a new set of images by evolving the rated images using genetic algorithm (GA). This human-in-the-loop exploration process is iterated until he/she finds a good result.

Although GA is good at effectively finding a global optimum solution, it sometimes prevents the user from exploring wide varieties of solutions. The exploration of diversified solutions is very important in the initial design process of

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 459–469, 2007.
© Springer-Verlag Berlin Heidelberg 2007

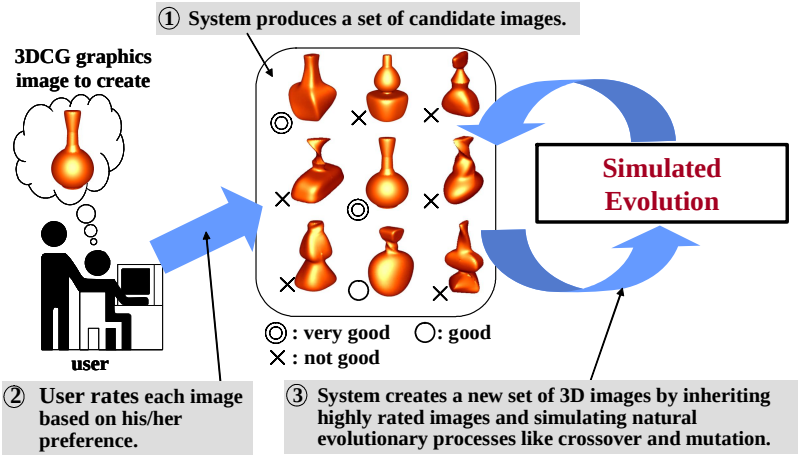


Fig. 1. Intuitive 3DCG authoring method based on human-in-the-loop IEC framework

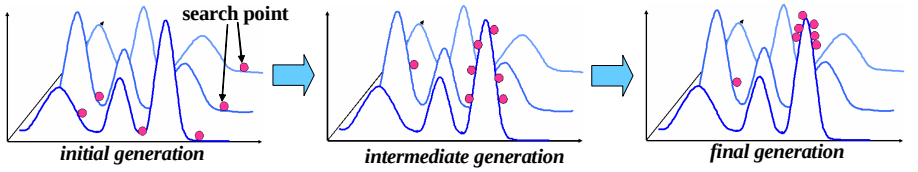
3DCG contents authoring. Consequently, many of the quasi-optimum solutions are still good candidates for a final 3D product. GA enables multiple search points to quickly come close to the global optimum solution (a highest peak) in the multi-peaks search space as shown in figure 2(a). As illustrated in the figure, a quick convergence referred as "premature convergence" discourages the user from widely looking for other good but not best solutions.

In this paper, we propose to apply immune algorithm (IA) as a core evolutionary computation technique to easily acquire wide variety sets of quasi-optimum solutions for the 3DCG contents authoring tasks. As shown in figure 2(b), IA can effectively explore multiple solutions by the aid of its intrinsic mechanisms such as antibody formation and self-regulating function.

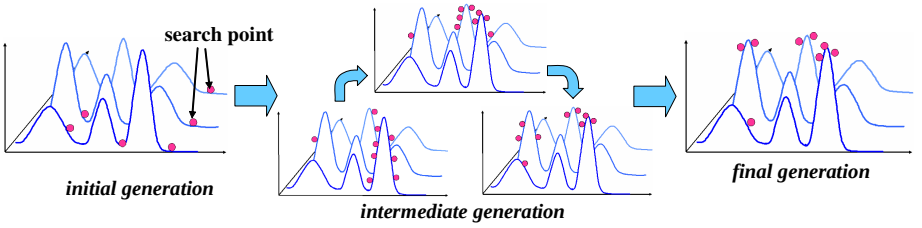
2 Related Work

IA was first proposed by Mori et al. [4] and has been proven to be a useful method to solve the multi-optimization problems. Among a number of methods proposed for the multi-optimization problems, IA can effectively search quasi-optimum solutions with smaller population size [5]. Because IEC requires the user to rate all candidate solutions (individuals) one by one, human fatigues caused by excessive operations need to be avoided [6]. Reducing the population size to a maximum extent is a crucial requirement to realize a successful IEC system. IA is a very good method to meet the goal.

There are several trials to apply IA to the multi-optimization problems. Toma et al. used IA to optimize multimodal functions [7] and n-TSP [8]. Nakayama et al. proposed a new quantum computing algorithm by embedding IA's antibody formation mechanism [9]. These existing systems require a predefined evaluation function and some threshold settings to automatically control the IA operations. Appropriate definition of these parameters is a crucial task to get multiple solutions,



(a) Optimization by GA: multiple search points come close to a highest peak in an early generation by "premature convergence" and finally find a global optimum solution.



(b) Optimization by IA: multiple search points can effectively explore a set of quasi-optimum solutions through its intrinsic mechanisms such as antibody formation and self-regulating function.

Fig. 2. Comparison of optimization in a multi-peaks search space by GA and IA

but it is a very tricky part to successfully use it. The proposed system improves the algorithm by adding some functions to interactively control the IA operations.

3 Human Immune System Overview

Figure 3 shows a human immune system overview. Two intrinsic mechanisms such as antibody formation and self-regulating function characterize the human immune system. The antibody formation is to produce and propagate antibodies to get rid of any unknown antigens. When the *antibody-producing cell*, a type of B-cell, detects an invading antigen, it produces effective antibodies by iterating genetic operations such

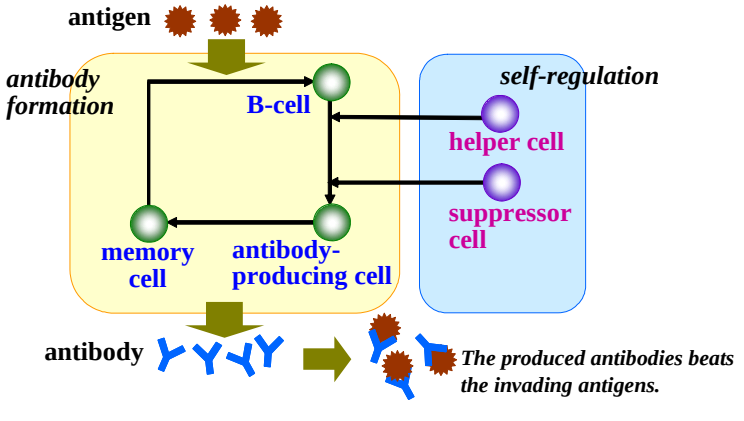


Fig. 3. Human immune system overview

as crossover and mutation over the existing antibodies. The *helper cell* accelerates B-cell production for efficient antibody formation. Once the effective type of antibodies to eliminate the detected antigen is produced, the *memory cell* memorizes a part of the produced antibodies. This memory mechanism that is referred to as "acquired immunity" quickly protects the human body from the invasion of the same antigen in the future. This is a very important mechanism to protect human body from catching diseases such as measles and mumps.

When the antibody formation excessively produces the antibodies to beat the invading antigens, the self-regulating function is activated to inhibit the growth of the antibodies. The *suppressor cell* deteriorates the B-cell production to return the immune system status to a steady state.

To apply the human immune mechanism to the IEC-based graphics authoring system, we assume that the antigen corresponds to an optimum solution (a final graphics image made by the user) and the antibodies are equal to candidate solutions (candidate 3D graphics models to evolve).

4 Interactive Immune Algorithm for 3DCG Authoring

As described in section 2, traditional IA algorithms require a predefined evaluation function to automatically calculate the fitness values of antibodies. They also need some preset threshold parameters to timely accelerate or suppress the antibody formation. Appropriate definition of these function and threshold values is a crucial problem to successfully acquire multiple solutions. The tuning of these values, however, is a tricky task and makes IA to be a difficult method to use for the multi-optimization problems. The proposed system improves the traditional algorithms by adding some functions to interactively control the acceleration or suppression of the antibody formation. The proposed interactive IA algorithm is described as a flowchart in figure 4. The shaded steps such as processes (b), (c), and (d) in the flowchart require user interventions to control the IA processes. Detail explanations of each process (a through h) in the flowchart follow:

(a) Creation of an initial generation
Firstly, IA creates an initial generation of antibodies by randomly calculating all parameter values encoded in each antibody's chromosome as shown in figure 5. If there are memory cells kept in the SMC-DB (suppressor and memory cell database), some cells are selected to compose the initial generation. Because the SMC-DB keeps optimum solutions discovered in the past trials in process (h), IA reuses such previously found solutions as good candidates to perform a new trial.
(b) Judgment on convergence status
A user specifies whether he/she finds an optimum solution, a 3D model (an antibody) that coincides with his/her imagery, in the current generation. He/she picks the found solution when exists and proceed to process (h), otherwise continues to process (c) for executing further simulated evolutions.
(c) Judgment on IA completion
The user indicates to finish the IA-guided 3DCG model exploration if he/she gets enough variety of the 3DCG models (a set of optimum solutions).

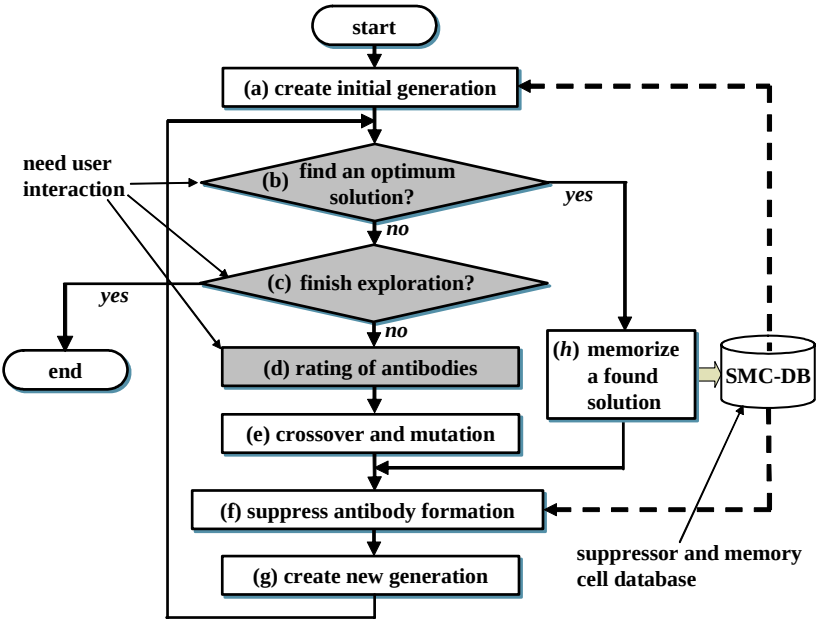


Fig. 4. Flowchart of interactive IA

(d) Rating of antibodies

The user rates each model (antibody) with his/her subjective preference on a scale of 1 to 5 (the worst to the best corresponds to 1 to 5). Each antibody's rate corresponds to a fitness value representing its degree of similarity with the antigen (user's final image to create). The highly rated antibodies, therefore, are similar to the antigen and have high expectation to survive in the future generations. The degree of similarity between the antigen and the antibody v is defined as follows:

$$ax_v = Fitness_v. \tag{1}$$

where $Fitness_v$ is a fitness value of the antibody v specified by the user.

(e) Crossover and mutation

This process selects a pair of antibodies as parents and performs a crossover operation on the pair to produce a new pair of antibodies (children). It uses each parent's fitness value as an expectation for the selection, so the highly rated antibodies have higher probability to be chosen as the parents. It also induces a mutation of gene to preserve a diversity of antibodies.

(f) Suppression of antibody formation

This process suppresses all child antibodies that are similar to the previously found optimum solutions. The purpose of the suppression is to keep the evolving antibodies away from search fields near the already acquired solutions. It enables to efficiently explore other unknown solutions in the huge search space. The previously found solutions are kept as the suppressor cells in the SMC-DB. Consequently, the suppression mechanism calculates the degree of similarity between each child

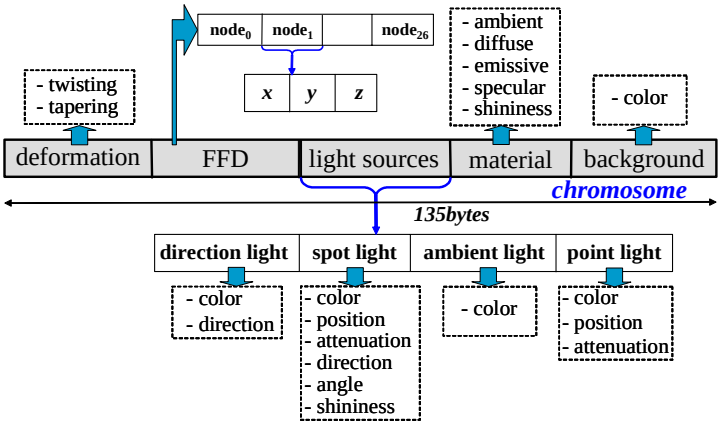


Fig. 5. Structure of a chromosome

antibody produced in process (e) and all the suppressor cells, and then suppresses all children whose similarity degrees are higher than the threshold value, the only predefined value in our algorithm. The degree of similarity between the suppressor cell *s* and the antibody *v* is defined as follows:

$$ay_{v,s} = 1 - \frac{\sum_{p=1}^{P} \sqrt{(g_v - g_s)^2}}{P} \quad . \tag{2}$$

where P is a population size, g_v and g_s are genes of the antibody and the suppressor cell, respectively. The g_v and g_s are corresponding genes and take real numbers between 0 and 1. Accordingly, $ay_{v,s}$ becomes 1, a maximal value, when the chromosome of the antibody is identical to the suppressor cell's.

(g) Creation of a new generation

This process produces additional antibodies by randomly setting the parameters if there are suppressed ones in process (f). The processes from (e) to (g) guarantee the exploration of a new solution (antibody) from undiscovered search field.

(h) Memorization of an optimum solution

This process memorizes the discovered optimum solutions as the memory cells in the SMC-DB. The stored solutions are reused as effective antibodies to form the initial generation in process (a). Because the SMC-DB can only store a limited number of memory cells, a replacement algorithm is activated when a cell pool in the SMC-DB fills. It calculates the degree of similarity between the found solution (antibody) and all the memorized cells by using the equation 2. Then, it replaces the most similar memory cell by the newly found solution.

The discovered solutions are also memorized as the suppressor cells in the SMC-DB. They are used to suppress the evolving antibodies that are similar to the already found solutions in process (e).

5 Interactive Graphics Authoring System

We designed and developed an IEC-based 3DCG authoring system that allows even a novice user to intuitively create 3DCG contents. As shown in figure 6, the system consists of two software components, the *IEC browser* for exploring 3DCG contents and the *I(individual)-editor* for manually elaborating CG parameters. Firstly, the system needs an initial 3D model to activate the IEC-guided authoring. The user has three options to prepare the model, (1) captures a real object (a vase example in figure 6) by using a 3D scanner, (2) produces it by using the freehand sketch modeler [10], or (3) retrieves it on the Internet. Next, the user invokes the IEC browser and initializes it with the initial model. The IEC browser controls the IA-based 3D model generation and simultaneously displays up to twenty images (antibodies). It allows the user to browse all candidate images in a screen and rate each image with his/her subjective preference on a scale of 1 to 5 (the worst to the best corresponds to 1 to 5).

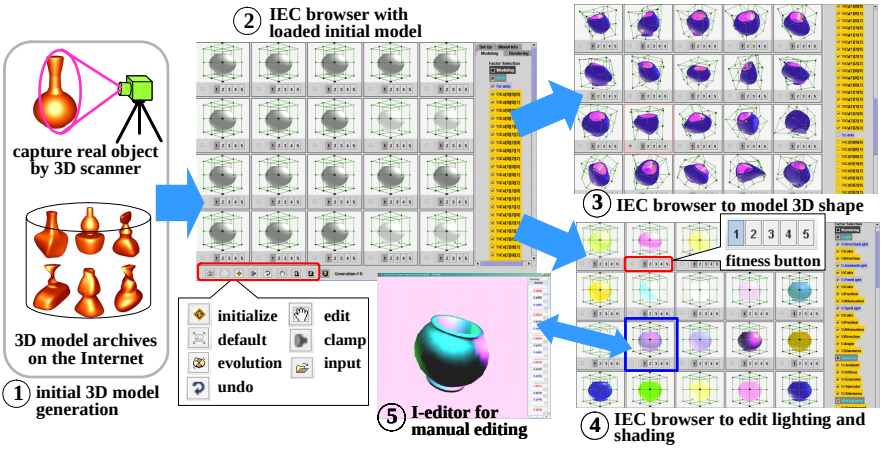


Fig. 6. IEC-guided graphics authoring system: an example of vase shape modeling

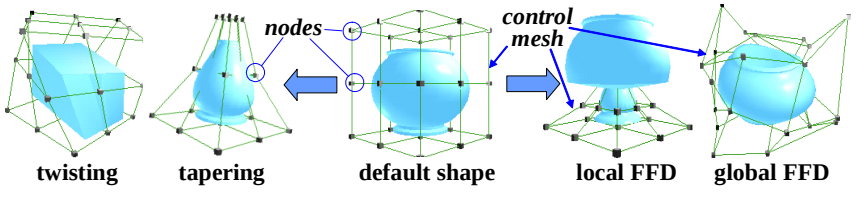


Fig. 7. 3D geometric modeling function implemented based on FFD

The I-editor provides a fine-tuning option to manually elaborate the graphics parameters of a candidate model. The user clicks a specific image's sub-window in the IEC browser to activate the I-editor. Then, he/she checks and modifies the parameter values that are controlling the selected model's color, geometrical shape, deformation patterns, surface materials, and lighting effects. Because the model image

drawn in the I-editor is immediately updated when the user modifies any parameters, he/she easily finds the effect of the changes and perceives his/her preferred parameter settings by manually changing the parameter values. After the manual edit, the modified object can be brought back to the IEC browser for further evolutions.

The system employs a modeling method called free-form deformation (FFD) [11] to perform 3D geometric operations on the initial model. As illustrated in figure 7, FFD wraps the target object with a simplified control mesh. When the mesh shape is changed by moving its nodes, the wrapped 3D object is deformed according to the modified mesh shape. The shape deformation can be performed globally (global FFD) or partially (local FFD) as shown in figure 7. A control mesh consists of 27 nodes (3x3x3 mesh) to evolve the object shape. All 27 nodes' positions are encoded as a chromosome to deform the mesh via the IA operations as shown in figure 5. The system supports tapering and twisting operations to implement a deformation method like the clay modeling as shown in figure 7.

The system also supports some rendering functions to represent the object's surface materials, lighting effects, and colors. As shown in figure 5, our system supports four types of light sources (direction, spot, ambient, and point lights) with a set of parameters to describe the object's surface materials and its background color. Figure 5 shows the structure of a chromosome that represents a 3D model evolved in the system. A gene is 8 bits long to encode each parameter and the total length of a chromosome including all modeling and rendering parameters is 135 bytes long.

As the evolution progresses, some parameters in the chromosome such as the object's shape or its surface material might be well converged and they need to be protected from further modifications. Therefore, the system supports a lock/unlock mechanism for each parameter to preserve a well-evolved part of the model. The system is written in Java to make it be downloadable on the Internet and be usable under any type of operating systems.

6 Evaluation Experiment

We conducted an experiment to verify the effectiveness of the proposed IA-based 3DCG authoring method. To examine how it can intuitively support the 3D modeling tasks, we prepared three different images of indoor places as shown in figure 8 such

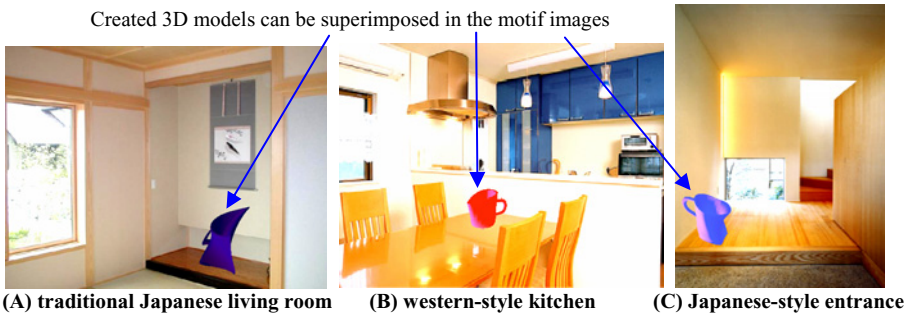
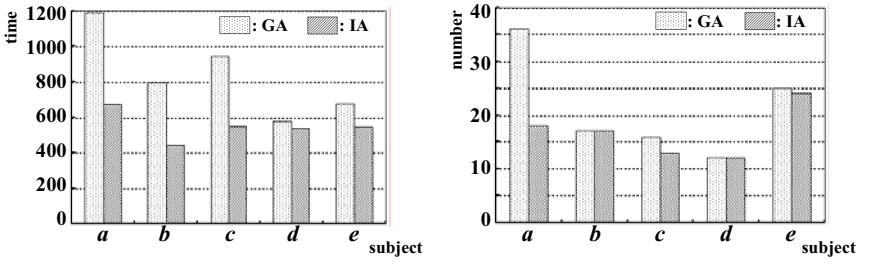


Fig. 8. Motif images of three indoor places used in the experiment



(a) total modeling time (second) (b) total number of generations

Fig. 9. Comparison of modeling performance between GA and IA

as (A) traditional Japanese living room, (B) western-style kitchen, and (C) Japanese-style entrance. Then, we asked five subjects to create a 3D cup model fitting in each image. To compare the IA-based method with the traditional GA-based one, we also requested the subjects to create two models for each image, which are one model by the IA modeler and the other by the GA modeler. Therefore, each subject produced six models in the experiment in total (IA and GA models for three motif images). Because the GA method can only find a single (optimum) solution in a single run, the subjects needed to run the GA modeler three times to complete the three motifs. On the other hand, the IA method can discover three solutions in a single run. After the subjects completed all tasks, they are asked to compare and evaluate both methods from the following viewpoints:

- Usability: which method is easier to use, and
- Satisfaction level: which method is better to get a satisfactory result.

All modeling tasks were performed under the following settings: population size is 16, crossover rate is 90% (remaining 10% individuals are inherited as elites in the new generation), mutation rate is 1%, and one-point crossover is used to create a new generation. In the case of the IA modeling, the threshold value to suppress antibody formation is 0.76 (the additional parameter to use IA as described in section 4(f)).

The graph in figure 9 (a) shows the total modeling time measured in the experiment on the second time scale. It indicates the total runtime spent for each modeler (IA and GA). The horizontal labels *a* through *e* denote the subjects. The subjects *a* and *b* rated that IA was much better than GA in the comparison as described above, *c* and *d* rated that IA was slightly better, and *e* answered that no difference was observed. No one supported the GA method. As found in the graph, the IA supporters (a, b, and c) finished the IA modeling quicker than the GA one. Though they only spent less than 60% of the GA's total time, they acquired better results in the IA case. The graph in figure 9 (b) shows the total number of generations counted during the whole modeling tasks. As observed in this graph, all subjects except *a* performed the nearly equal number of evolutions in both cases to complete the experiment. Accordingly, the proposed IA method allowed the subjects to efficiently iterate the evolutions and produce better results in a shorter amount of time.

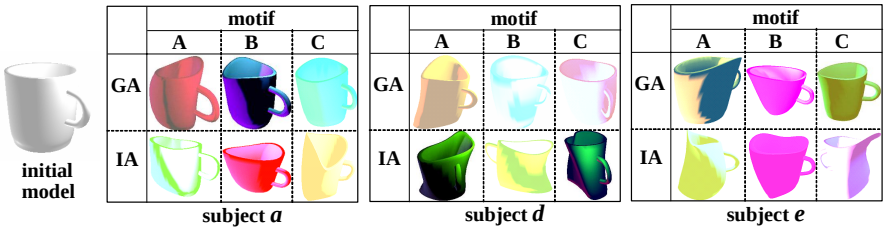


Fig. 10. 3D models produced in the experiment

Figure 10 shows the 3D models produced by three subjects who gave different ratings in the comparison. The leftmost one is an initial cup that is a typical shape with no color (white). The subjects created the final models by changing the color and shapes of the initial model through the evolutions. Whereas the models made by the subjects *a* and *d* who supported the IA method are quite different in color and shape between IA and GA, the models produced by the subject *e* who rated an even score turn out to be similar results. The IA supporters emphasized the IA method's intuitiveness and expressive power for the 3D modeling in their introspective reports, but the subject *e* mentioned that he couldn't see any difference on both methods.

Because the number of subjects is not enough to statistically prove the significance of the proposed IA method, we would like to continue the evaluation to improve the reliability of the experiment.

7 Conclusions

We proposed an approach to easily explore a variety of 3D graphics models based on IA (immune algorithm). It provides an intuitive way to create various 3D models with different impressions. We implemented a 3DCG authoring tool based on the proposed interactive IA algorithm. The experiment presents that the proposed IA method can produce better graphics solutions with maintaining the multiplicity of evolved candidates (antibodies) over the traditional GA-based method. As a future work, we would like to conduct more detailed experiments to perform a deeper analysis and statistically prove the effectiveness of the proposed method.

References

1. Nishino, H., Aoki, K., Takagi, H., Kagawa, T., and Utsumiya, K.: A Synthesized 3DCG Contents Generator Using IEC Framework. Proc. of the IEEE SMC'04, pp.5719-5724, 2004.
2. Nishino, H., Takagi, H., and Utsumiya, K.: A 3D Modeler for Aiding Creative Work Using Interactive Evolutionary Computation. Trans. of the IEICE, Vol.J85-D-II, No.9, pp.1473-1483, 2002 (in Japanese).
3. Takagi, H.: Interactive Evolutionary Computation: Fusion of the Capabilities of EC Optimization and Human Evaluation. Proc. of the IEEE, Vol.89, No.9, pp.1275-1269, 2001.

4. Mori, K., Tsukiyama, M., and Fukuda, T.: Immune Algorithm with Searching Diversity and its Application to Resource Allocation Problem. Trans. IEE Japan, Vol.113-C, No.10, pp.872-878, 1993 (in Japanese).
5. Mori, K., Tsukiyama, M., and Fukuda, T.: Application of an Immune Algorithm to Multi-Optimization Problems. Trans. IEE Japan, Vol.117-C, No.5, pp.593-598, 1997 (in Japanese).
6. Ohsaki, M., Takagi, H., and Ohya, K.: An Input Method Using Discrete Fitness Value for Interactive GA. J. Intelligent and Fuzzy Systems, Vol.6, pp.131-145. 1998.
7. Toma, N., Endo, S., and Yamada K.: An Adaptive Memorizing Immune Algorithm for Multimodal Functions. IEICE Tech. Report, Vol.99, No.539, pp.71-76, 1999 (in Japanese).
8. Endo, S., Toma, N., and Yamada K.: Immune Algorithm for n-TSP. Proc. of the IEEE SMC'98, pp.3844-3849, 1998.
9. Nakayama, S., Ito, T., Iimura, I., and Ono, S.: Proposal of Mixed Interference Crossover Method in Immune Algorithm. Trans. of the IEICE, Vol.J89-D, No.6, pp.1449-1456, 2006 (in Japanese).
10. Nishino, H., Takagi, H., Saga, S., and Utsumiya, K.: A Virtual Modeling System for Intuitive 3D Shape Conceptualization. Proc. of the IEEE SMC'02, Vol.4, pp.541-546, 2002.
11. Sederberg T. W. and Parry S. R.: Free-Form Deformation of Solid Geometric Models. SIGGRAPH '86 Proceedings, pp.151-160, 1986.

Human Mosaic Creation Through Agents and Interactive Genetic Algorithms Applied to Videogames Movements

Oscar Sanjuán[1], Gloria García[1], Yago Sáez[2], and Cristobal Luque[2]

[1] Universidad Pontificia de Salamanca.
Paseo de Juan XXIII, nº3. 28044. Madrid
{gloria.garcia,oscar.sanjuan}@upsam.net
[2] Universidad Carlos III de Madrid.
Av. de la Universidad, nº20. 28911, Leganés, Madrid
{yago.saez,cristobal.luque}@uc3m.es

Abstract. In this essay the construction of an interactive genetic algorithm and agents based application will be described. This construction is driven to plan the development of a human mosaic, starting from an initial design and an adjustment of several parameters. In addition, the same idea of generation will be show as suitable to be applied to character group movements inside a videogame. The software developed is enclosed, as far as the "creative" part of the solution is concerned, inside the evolutionary computation paradigm; while into agent orientation regarding characters coordination.

Keywords: Genetic algorithm, software agents, interactive genetic algorithm, videogames, human mosaics, group movements.

1 Introduction

The first question to be answered is: what a human mosaic is? A human mosaic can be a musical choreography, which is leaded to achieve an artistic representation of an image, symbol, or anagram, by the coordination of a group of people that will perform the mosaic composition. The most common example is the human mosaics represented in the Olympic Games (see figure 1).

Why design squad game movements through a human mosaic? Precision in the real representation is the goal to be reached, not only in the characters' own behavior but also in motion itself (individual or even collective) [1],[2]. The aim is to give reality to the game. Acting this way, immersion into the created world will be achieved.

The solution focus will be developed by using two techniques: the system agents, and the genetic algorithms [3].

First of all, the selected strategy to face the problem will be presented, and then the must of using genetic algorithms and agent orientations paradigm will be justified. Section five covers briefly this subject's state-of-art, while point six develops a hands-on approach in order to help the problem to be understood. Later on, the solution adopted is depicted, as well as how it can be applied to a videogame when group movements are needed (i.e. a Role Playing Game). To conclude, final considerations and possible future developments are presented.

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 470–476, 2007.
© Springer-Verlag Berlin Heidelberg 2007

2 Solving Problem Strategy

The problem presented here has three faces: first, the creative process, the possibility of this process to be mechanized (this process is widely covered in [4]). The second face to deal with is the route creation. And finally, an optimal set of parameters must be used in order to perform well.



Fig. 1. Human Mosaic

3 Interactive Genetic Algorithms in Mosaic Creations

Interactive Evolutionary Computation (IEC) is an optimization method, which adopts Evolutionary Computation (EC) programming inside an optimization system based on human subjectivity [5]. In 1991, Karl Sims showed how genetic algorithms are able to use 'fitness functions', to procure complex and beautiful images using human subjectivity as their basis [6]. The use of interactive genetic algorithms can find, while coordinated with the user, a potential solution for a large set of complex problems, [5],[7].

Mosaic creation is both a creative and design problem, which has no a single solution; actually, it depends on subjective parameters. The human intervention for this problem is required, because only the user can determine which group of movements is the most adequate for him o her needs. Thus, the user active participation inside the optimization problem is crucial and, consequently, IEC and more precisely Interactive Genetic Algorithms (IGA's) have been adopted as the solution paradigm.

4 Agents Applied to the Problem

Inside a computer system, an agent placed in an appropriate environment is capable of behave in an autonomous way to achieve its own goals [3]. Each of the parties inside



Fig. 2. Black and White



Fig. 3. Unreal

the mosaic will act as an agent inside our system, with its particular behaviour, and, in addition, with its unique genetic characteristics (as it will be later presented).

The agents-oriented focus will allow complex interactions among the population of the individuals. As soon as the model is converted to a videogame, the individuals (the ones that are involved in the movement which could be performed) can interact in their particular way influenced by their own characteristics and the environment in which they are inside.

Inside videogame ambit, agent and adjacent technologies (usually related to I.A.) are often used. Nowadays videogames like “blackandWhite” (Figure 2) or “EnemaNaions”, have been developed using agents, there are others like “unreal” (Figure 3) that have used related concepts like bot (www.gameai.com). Recently there

are projects that use agents in order of facilitate the videogame design, an example is the project leaded by M.I.T (http://agents.media.mit.edu/projects/videogame).

The use of agents facilitate e the task of coordination thanks to the development of coordination language ACOL (agent coordination language) based on ACL (agent communication Language), which belongs to FIPA's standard) and XML language. ACOL is actually under development.

5 State-of-the Art. Preliminaries

Nowadays, there are a large number of architects and designers who use evolutionary techniques in their projects in order to create shapes, maps or environments, or different characters. It is also widely recognised the use of the interactive evolutionary computation in numerous subjects, as the image regeneration by an interactive genetic algorithm which dynamically improves an image [6], web systems style sheets generation based on user preferences [8], furniture design automatic generation [9], or even several well known "evo-art" examples. Nevertheless, as mentioned before, see Takagi [7] for a complete reference of IEC applications up to the date.

Another main antecedent inside parallel systems is the use of agents as Massive[1], software devoted to virtual agents generation applied in simulations or films (i.e. "Lord of the Rings"). Here is used to build up huge armies.

6 Previous Work

When the problem was first faced, a prototype was designed to validate the application of the genetic algorithms in human mosaics; now the problem to solve is how the interactive genetic algorithms with agents can be applied to squad movements in computer games. The first thing to do is choosing the main parameters of the process:

- Participant number.
- Surface that must be covered.
- Performer role (Elements of the mosaic which is going to be represented).

These parameters permit to start the evolving process and as soon as the performer's roles have been defined, the next step will be set the path of each individual in order to make them reach their own place inside the design.

To generate the different routes it is mandatory to establish their starting and ending points; to do so, these specific parameters must be marked:

- Initial position. Each character starts from an initial position, which can consist on a combination of these parameters:
    - Fixed / occasional.
    - Common / individual.
    - Inside / outside the surface.

[1] http://www.massivesoftware.com/ready_to_run_agent.html

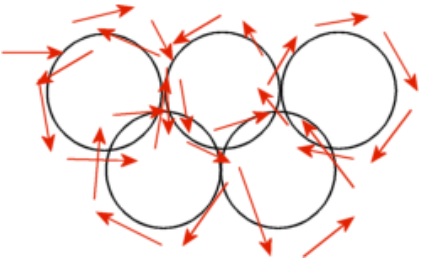


Fig. 4. Common position

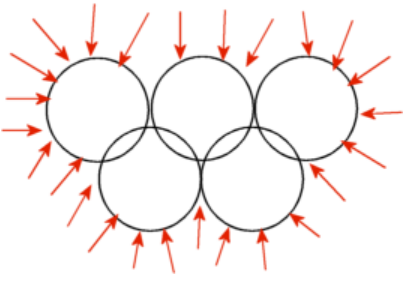


Fig. 5. Outside position

- Path
  - Single file. Characters join in a row to finish the mosaic.
  - Attack. Each character tries to fulfil the shortest path through its final position.

As an example, in a first approach (Fig. 4) the players share an initial common position outside the mosaic surface, and follow a single file path.

In a second approach (Fig. 5), each player has an individual initial position (random) outside the surface bounded to develop the mosaic, and its path is of the attack kind.

Finally, in the third approach all the players share a common occasional position inside the mosaic surface, this time their path is again ‘attack’ (Fig. 6).

In the strategy games it is very usual to have a specific group of population which is devoted to complete a task by placing them in the best way. It is important to

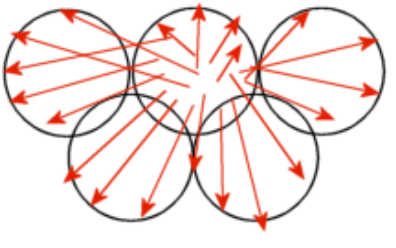


Fig. 6. Inside surface position



Fig. 7. Solution development example

analyze the resulting movements as well as the movement itself, so it is possible to find zones which can be dangerous for individuals.

7 Conclusions

The IEC problem wanted to solve is the movement strategies inside a videogame. In this creation process, human intervention determines which group movements are the most adjusted to the objectives. In addition, the combination of several movements to create a new one is allowed.

Once the algorithm has been developed and applied, the new goal is to optimize the general parameters so that the user can get satisfying movements within a few iterations. The algorithm parameter adjustments are themselves an optimization problem.
Dealing with IEC applied to small populations, two problems arise:

1. The individuals must perform their subjective evaluation and, in addition, the users are not likely to assign numeric values.
2. Because of the lack of massive genetic diversity, the algorithm tends to converge to similar results within a few generations.

There are several approaches which try to solve these problems, and the solution adopted is the use of a joint fitness [11] which helps in the use of micro populations. Joint fitness allows genetic diversity to grow, because population can be increased without the necessity of complicate user's performance.

The user will evaluate a subset within the population, which would have been previously filtered by an automatic evaluation function based both in user specific defined and system viability criteria.

As explained in [12], the users will be able to choose as much movements as they want to. This selection, combined with the implicit fitness, will shape the joint fitness that allows the system to generate a new set of solutions.

This prototype has been developed under C++, and the open source game Glest[2] (a military strategy game).

[2] www.glest.org

8 Future Developments

The IGA's can be classified as mechanisms that support creative and artistic processes and they can be used to find suitable strategies in role videogames.

Our next intention is to improve the actual model in a number of aspects:

- Improvement inside the agents system, in order to provide them with coordination and communication capacities, especially designed for strategy games.
- Improvements to user reality experience thanks to the genetic intelligence of the agents.
- A new parametric approach designed for role playing games.

References

[1] Gordon A. "Enabling and recognizing strategic play in strategy games: Lessons from Sun Tzu". University of Southern California, (2002).

[2] Sanjuán, O. et al. "Human mosaics creations through genetic algorithms and agents", V Congreso Español sobre Metaheurísticas, Algoritmos Evolutivos y Bioinspirados, MAEB'2005. (2005).

[3] Sanjuán O. "Métodos evolutivos para la construcción de agentes adaptativos en entornos cambiantes heterogéneos", PhD thesis dissertation, Universidad Pontificia de Salamanca, (2006).

[4] Bentley P. An Introduction to Evolutionary Design by Computers. (1999).

[5] Takagi H. "Interactive Evolutionary Computation: Fusion of the Capabilities of EC Optimization and Human Evaluation". Proc. of the IEEE. Vol. 89, 9, pp. 1275-1296. (2001).

[6] Kato S., "An image retrieval method based on a genetic algorithm controlled by user's mind", Journal of the CRL, Vol. 48, No. 2, pp. 71-86, (2001).

[7] Takagi H. "Interactive Evolutionary Computation: System Optimization Based on Human Subjective Evaluation". International Conference on Intelligent Engineering Systems (INES'98). Vol. 89, 9, pp. 1-6. (1998).

[8] Monmarche N., Nocent G., Slimane M.. Venturini G. and Santini P., "Imagine: a tool for generating HTML style sheets with an interactive genetic algorithm based on genes frequencies", Proc. IEEE Int. Conf. on Systems, Man and Cybernetics, IEEE Press, Piscataway, NJ. pp. 640-645, (1999).

[9] Sáez Y., Sanjuán O., Segovia J. "Algoritmos Genéticos para la Generación de Modelos con Micropoblaciones", Primera Conferencia de Algoritmos Evolutivos y Bioinspirados, (2002)

[10] Sáez Y., Sanjuán O., Segovia J., Isasi P., "Genetic Algorithms for the Generation of Models with Micropopulations", Proc. of the EUROGP'03, Univ. of Essex, UK., Springer Verlag, pp. 490-493, (2003)

[11] Sáez Y., Isasi P., Segovia J., Hernández J.C., "Reference chromosome to overcome user fatigue in IEC", New Generation Computing, vol. 23, num. 2, Ohmsha - Springer, pp. 129-142, (2005).

[12] Turau C. "Synthesizing movements for computer game characters". Universidad de Bielefeld, Germany, (2004).

Self-organizing Bio-inspired Sound Transformation

Marcelo Caetano[1], Jônatas Manzolli[2], and Fernando Von Zuben[3]

[1] IRCAM-CNRS-STMS 1place Igor Stravinsky – Paris, France F-75004
[2] NICS/DM/IA - University of Campinas – Brazil PO Box 6166
[3] LBiC/DCA/FEEC – University of Campinas – Brazil PO Box 6101
caetano@ircam.fr, jonatas@nics.unicamp.br,
vonzuben@dca.fee.unicamp.br

Abstract. We present a time domain approach to explore a sound transformation paradigm for musical performance. Given a set of sounds containing *a priori* desired qualities and a population of agents interacting locally, the method generates both musical form and matter resulting from sonic trajectories. This proposal involves the use of bio-inspired algorithms, which possess intrinsic features of adaptive, self-organizing systems, as definers of generating and structuring processes of sound elements. Self-organization makes viable the temporal emergence of stable structures without an external organizing element. Regarding musical performance as a creative process that can be described using trajectories through the compositional space, and having the simultaneous emergence of musical matter and form resulting from the process itself as the final objective, the conception of a generative paradigm in computer music that does not contemplate *a priori* external organizing elements is the main focus of this proposal.

1 Introduction

The digital computer allows great flexibility in sound processing. As a consequence, the spectrum of possibilities is so vast that the exploration of the full musical potential of the digital computer has become a major problem in computer music. Many different approaches have been proposed in order to create aesthetically interesting music for composition and performance, with results that vary from the unexpected to the undesired, depending upon a vast number of factors and on the methodology itself [7]. Musical sounds are complex and hard to synthesize, since they generally have a dynamic spectrum, such that each partial frequency has a unique temporal evolution envelope. Our ears are highly selective and frequently reject mathematically perfect and stable sounds [7]. Traditional sound synthesis techniques are limited, especially because they do not take the dynamic and/or subjective nature of music into consideration, using deterministic processes that were not specifically designed for sound manipulation [9].

In the perspective of this work, music composition and performance are creative processes that can be described using trajectories through soundspace that create musical matter and form by means of a self-organizing generative and organizational process. Self-organizing systems exhibit emergent properties and promote complex

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 477–487, 2007.
© Springer-Verlag Berlin Heidelberg 2007

patterns from simple rules. Here, self-organization is the principle used to generate, organize and structure sound material during musical performance, guiding the composer through the compositional soundspace.

The use of bio-inspired or AI based systems for musical composition, performance, and free improvisation has already been suggested independently by several researchers as a way to permit more flexibility in the aesthetical exploration of the resulting compositional space. The applications of bio-inspiration in composition involve artificial neural networks [11], cellular automata [6], artificial immune systems [7], particle swarms [2], ant algorithms [14], and evolutionary computation [1]. Nevertheless, most of these systems use MIDI codification (i.e. representation of musical events rather than musical sounds), consequently, they generate only form using bio-inspired algorithms. On the other hand, Miranda's CAMUS [17] and Blackwell's Swarm Granulator [3] extract parameters from the agents allowing the generation of matter by means of granular synthesis, as well as form, represented as the dynamic state of the system. Independent works have proposed the use of genetic algorithms [8], artificial neural networks [9], and artificial immune systems [7] in a sound synthesis technique in time domain, highlighting the profitable aspects of each. The user is enabled to find candidate solutions that meet certain musical requirements by using a set of waveforms (attractors) as examples of the desired sound qualities, instead of describing the sounds using numerical parameters or any other linguistic tool. Developing this idea one step further, we are proposing here that in self-organizing systems with emergence of sound patterns, the synthesis process is integrated into the performance process so that as the system generates emergent patterns, this material constitutes sonorities for musical performance. By regarding self-organizing bio-inspired algorithms under the general umbrella of self-organization, the method allows the composer to express a certain degree of subjectivity by simply choosing the algorithm and setting the parameters adequately, according to aesthetical preferences that can be reflected in the particularities of each candidate paradigm.

The next section explains briefly the conceptual foundations of our method, followed by an introduction to the bio-inspired paradigms chosen to explore the compositional space in this study, trying to emphasize that although they all share the same high level organizational principle, that is, self-organization, the application of different approaches results, in the case of this work, in different trajectories through the space, and therefore, in different sound transformation procedures. This feature of the proposal is exemplified in the following section, where an example is used to illustrate the differences and similarities that are considered most relevant. Finally, the conclusions and future perspectives are considered.

2 Conceptual Foundations

In this section we present general ideas behind our model that lay the foundations of the populational search through soundspace in time domain. Here we show how time-domain codification results in a search space that approximates the sonic continuum. The aim of this section is to show that trajectories in the search space correspond to musical gestures, or manipulations of the sonic continuum.

2.1 Sound Transformations as Trajectories

At first, in traditional western music, the usual musical parameters are: frequency, time, intensity, and timbre. Often, these parameters are traditionally described with discrete values. Following this idea, musical composition can be understood as the organization of a finite set of notes, rhythms, timbres and different intensities using a summative notation (compass) and fixed durations [21]. Our approach is an attempt to escape the traditional musical lattice. The use of the computer as a musical instrument permits to go beyond the discrete, not being limited by its finite set of possibilities, reaching the sonic continuum and the transformation concept. A trajectory in this space represents a gradual transformation in one or more musical dimensions, that is, a gesture [21]. In this space, frequency is a real variable, as is time and intensity. Traditionally, timbre is described by a discrete set of values, since each different acoustical instrument characterizes a specific timbre and there is no middle point between two or more instruments. However, in the continuum it is possible to obtain sounds with characteristics originated from multiple instruments by just setting a position between two or more instruments in the dimensions representative of their timbral qualities.

2.2 Self-organization as Compositional Paradigm in Computer Music

Self-organization can be summarized as a process where a global pattern emerges from multiple interactions of simple system components. In this sense, the rules that specify components interactions only use local information, without any reference to a global pattern. It is in the emergence of structures and organization that are not imposed from outside of the system and that were not pre-established that lays the essence of self-organization [10]. In self-organized systems, properties that cannot be understood by examining the properties of the system's components arise from interactions among them [20]. The decision-making and emergent pattern processes of these systems are often non-intuitive due to the large number of nonlinear interactions involved. For this reason, mathematical models and computational simulations provide useful techniques for the study of self-organizing systems and for the exploration of the consequences of the myriad interactions among component subunits. Many of these simulations are agent-based, i.e., each individual subunit is monitored during the simulation, and its behavior over time is determined through local interactions with other subunits and local cues from the environment. This is precisely the case of this work, where each sound wave characterizes an agent from a population searching through soundspace. Agents interact locally and it is the resultant set rather than isolated individuals that represents a particular region of interest of soundspace, characterized by the attractors.

2.3 Codification and Resultant Search Space

Figure 1 illustrates the codification used in this work and the associated transformational mappings involved. Each individual (agents and attractors) is codified as N samples of a waveform sampled at a rate SR, as shown in part a). Each of these samples is interpreted as a vector component. Therefore, each individual is a vector with N dimensions. Part b) represents the two-dimensional Euclidean space

with a pictorial example of the distribution of a set of sounds interpreted as vectors in this space. This is only an illustrative example in which N=2. The sounds are manipulated by the algorithms in this space. There is a corresponding mapping to soundspace since each sound has duration, pitch (for harmonic sounds), timbre and dynamics. The generative process is understood as manipulations of this space, i.e., trajectories in the Euclidean search space correspond to articulations in one or more dimensions in the correspondent continuous sound space.

Smalley [19] declared that the information contained in the frequency spectrum cannot be separated from the time domain since "*spectrum is perceived though time and time is perceived as spectral movement*". So, once the user specifies the waveforms (attractors), he is also specifying the spectral content and sound qualities of the tones. Our objective is the attainment of results that correspond to gestures through temporal representation and manipulation of tones in order to allow musical performance. Each bio-inspired approach selected to guide the search through Euclidean space does so in a different fashion. They all share some basic characteristics, though, as follows. The agents of the system and the attractors characterizing the space belong to the same vector space, i.e., they have the same number of samples (dimensions), which corresponds to the same maximum duration.

The agents are initialized in points in the space that do not correspond to the attractors, meaning they are different sounds. Upon starting the simulation, the agents are drawn towards the regions containing the attractors. When one follows the trajectory described by each individual agent in pursuing the attractors, it can be perceived as a gradual transformation from the starting sound to the attractor (also a sound), drifting according to the bio-inspired approach, thus characterizing a unique gesture. Also, upon convergence, each approach tends to place the agents in different positions relative to the attractors when attempting to preserve certain features of the original space, i.e., the attractors themselves, resulting in variations of the sounds represented by the attractors.

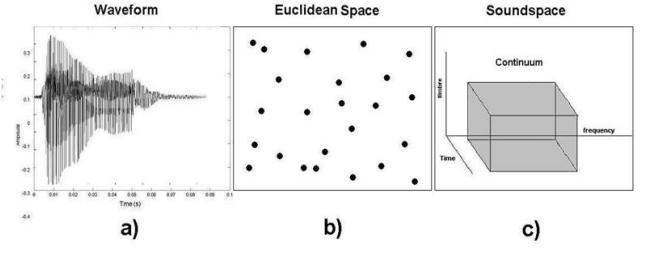


Fig. 1. Codification, pictorial resulting space and mapping to soundspace. In part a) there are N samples of a sound wave. Each waveform can be interpreted as a vector with N dimensions. Part b) represents the two-dimensional Euclidean space and the distribution of a set of sounds interpreted as vectors in this space (N here is taken to be 2, but the sound vector may contain thousands of elements), to be manipulated by the algorithm. There is a mapping to soundspace (part c) since each waveform has duration, a definite pitch (harmonic sounds), timbre (represented as one-dimensional for simplicity) and intensity. The compositional process happens in this space.

3 Bio-inspired Algorithms

3.1 Artificial Immune Systems

The immune system is a complex of cells, molecules and organs with the primary role of limiting damage to the host organism by antigens. One type of response is the secretion of antibodies, receptor molecules with the primary role of recognizing and binding, through a complementary match, with an antigen. Antigens can be recognized by several different antibodies. The antibody can alter its shape to achieve a better match (complementarity) with a given antigen [12].

Artificial Immune Systems (AISs) are adaptive procedures inspired by the biological immune system for solving several different problems. The model used here, aiNet [13], follows some ideas from the immune network theory, the clonal selection [5], and affinity maturation principles. The resulting self-organizing system is an antibody network that recognizes antigens (input data set, in this case, the attractors) with certain (and adjustable) generality.

The antibodies generated by aiNet will serve as *internal images* (*mirrors*) responsible for mapping existing clusters in the data set (Figure 2a) into network clusters (Figure 2b). The resultant memory cells represent common features present in the data set that were extracted by aiNet. Let us picture a set of sounds as antigens and its internal (mirror) image as variants. Inspired by Risset's sound variants idea [18], it is possible to imagine, for example, variants as a type of immune-inspired transformation applied to the sound population. Wishart's [21] ideas on the manipulation of the sound continuum as a compositional procedure induce one to regard the convergence process of the algorithm in time as a gesture (sound transformation procedure). In this sense, the waveforms can be regarded as the repertoire to which the system is exposed, and the associated sound qualities may be linked to the specific response it elicits. It is of critical importance to notice that, upon system convergence, when an antibody-sound is representing more than one antigen-sound, it is placed in such a spot in soundspace that allows it to present features that are common to all the sounds it is representing. Figure 2c depicts the intersection of characteristics shared by three different sounds.

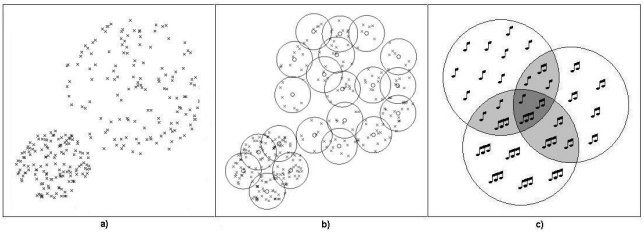


Fig. 2. Depiction of the representation of aiNet. Part a) shows the original dataset (attractors) in soundspace, part b) shows the resultant antibodies representing the attractors, and part c) illustrates the common timbral features of three classes of sounds.

3.2 Artificial Neural Networks

One important organizing principle of sensory pathways in the brain is that the placement of neurons is orderly and often reflects some physical characteristic of the external stimulus being sensed. Although much of the low-level organization is genetically pre-determined, it is likely that some of the organization at higher levels is created during learning by algorithms which promote self-organization. Kohonen [16] presents one such algorithm which produces what he calls self-organizing feature maps (SOMs) similar to those that occur in the brain.

Kohonen's algorithm creates a mapping of high-dimensional input data into output nodes arranged in a low-dimensional grid. The artificial neurons (output nodes) are extensively interconnected by many local connections with associated weights. The weights will be organized such that topologically close nodes (neurons) are sensitive to inputs (attractors) that are physically similar. The artificial neurons will thus be ordered in a natural manner [16].

There is no consensus ordering or classification for soundspace (Figure 3a). Due to the self-organizing feature of SOM, it is possible to propose arrangements that respect the original topology (Figure 3b). The key feature of SOM that allows this process is that attractors with similar characteristics trigger neurons in close regions of the one-dimensional mapping that represents the topological neighborhood in the original soundspace. In our application, self-organization gives rise to a musically profitable phenomenon. The result of the training may cause the neurons to represent more than one attractor (zoomed in areas in Figure 3b). The expected result is a merger of their qualities (Figure 3c). The concept of musical performance emerges from the possibility of following the dynamic convergence process of the neuronal sounds from the initialization to the final result. This process would reveal the neurologically induced transformation resulting from the path followed by each neuron during the self-organizing process. Moreover, the orderly self-organizing cyclic path provided by the method could also be advantageous.

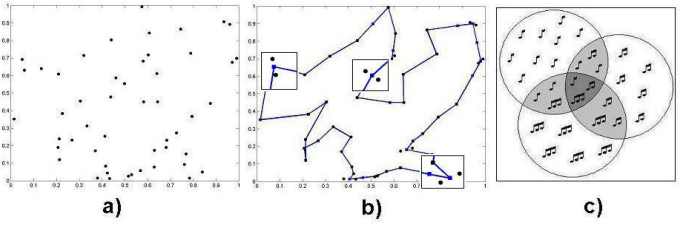


Fig. 3. Depiction of the representation of the one-dimensional SOM used. Part (a) shows the original topology of the space as represented by the waveforms (black dots) – again, an illustrative two-dimensional space is considered for visualization purposes; part (b) shows the resultant one-dimensional SOM representing the original space topologically arranged; and part (c) has the same meaning of Figure 2c.

3.3 Swarm Intelligence

Bonabeau et al. [4] define swarm intelligence as "*the emergent collective intelligence of groups of simple agents.*" While intelligence is usually considered to be a trait

possessed by humans alone, it can be shown to arise from interactions among individuals. Most species interact for adaptation purposes. Research in swarm intelligence is based on the grounds that there is a relationship between adaptability and intelligence, and that social behavior increases the ability of organisms to adapt.

Particle swarm optimization (PSO), the swarm intelligence paradigm used in this work, is a bio-inspired computer paradigm based on human social influence and cognition [15] whereby the particles interact to find the attractors, flying through the search space. The power of the particle swarm comes from the interactions of the individuals [15]. Individuals in the swarm have memory, they can remember the closest to the attractor they have been. Also, the particles are connected to other particles in a kind of social network. The particles it is connected to are called its neighbors. Each particle chooses the next point to visit by referring to its own previous best success and the best success of its best neighbor. In this approach, it is also influenced by the position of the nearest attractor. Thus the topological positioning of the individual particles relative to one another in the sociometric space has a profound effect on the swarm's ability to successfully find the attractors.

In this kind of representation the multidimensional psychometric model of human sociocognition, represented by our perception of the sounds, is nested in a kind of topological sociometric space (Figure 4a) that is comprised of the waveforms. Therefore, the self-organizing search for the attractors performed by the swarm corresponds to their approximation to the attractors in Euclidean space (Figure 4b), and the generation of sounds that share perceptual qualities in the cognitive evaluative space (Figure 4c). The way the particles interact affects the dynamics of the process, which, in turn, affects the trajectories (sound transformation procedures).

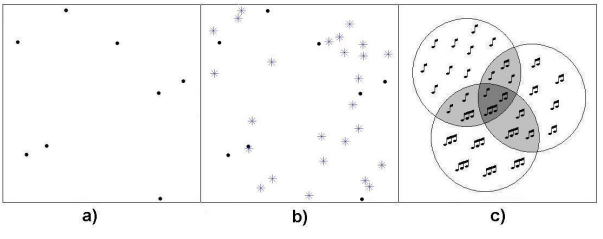


Fig. 4. Depiction of the capability of representing different regions of the environment owing to the cognitive exchange of information among individuals. Part (a) shows the original attractors as black dots, part (b) shows the particles as asterisks searching the space to best represent the original data, and part (c) has the same meaning of Figure 2c.

4 Results and Discussion

This section aims at illustrating the similarities and differences among the bio-inspired algorithms, trying to emphasize the musical applications of each. Each algorithm was run under very similar conditions, except for parameters that find no match in the other approaches, in order to highlight essentially the differences that are due to the distinct approaches. The experiment was performed as follows: seven attractors (different musical sounds) were used; each algorithm was initialized at

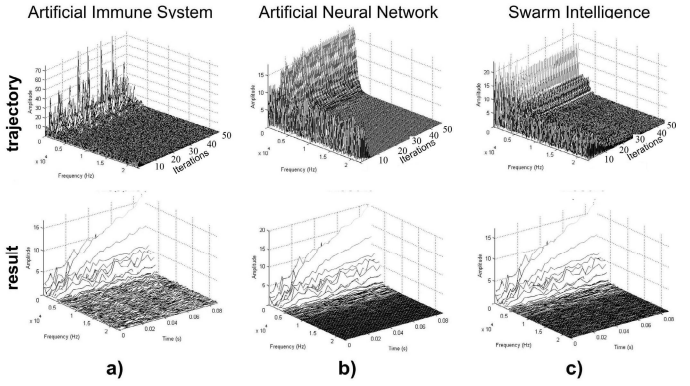


Fig. 5. Depiction of the different trajectories followed by the diverse bio-inspired algorithms over 50 iterations from white noise (random) initialization at the top row, and the resultant sounds at the bottom row. Part (a) shows one antibody-sound resulting from the application of aiNet, part (b) shows one neuronal-sound resulting from the application of Kohonen's self-organizing map, and part (c) shows one particle-sound resulting from the application of particle swarm intelligence. Each individual in the trajectory is represented just as the result, the only difference being the number of agents (50) shown at the top of the figure, to represent the entire trajectory followed by it.

random (white noise) with 14 agents and they were run for 50 iterations (discrete steps). Apart from aiNet, whose population varies dynamically, the others maintain the same number of agents over the iterations.

All algorithms converged before 50 iterations with at least one agent representing each attractor. When more than one agent moves toward one attractor, they do so through different paths, representing distinct sound transformations, specially because they were initialized in separate locations in Euclidean space, thus being different starting sounds.

At the top row of Figure 5, the trajectory of one of the agents for each algorithm is presented over all the iterations, as well as the final result (bottom row) achieved by each algorithm. All the agents chosen to be shown pursued the same attractor, which can be confirmed by visual inspection of the final result for each paradigm. The trajectory (top row) is represented as the agent at each iteration, whereas the agent is represented as its dynamic spectrum. Since each algorithm features diverse bio-inspired mechanisms to seek the attractors, different trajectories are expected to arise from the same conditions. Nevertheless, all algorithms are performing the same musical task: find the attractor and represent it following pre-specified criteria that vary accordingly. A different trajectory can be visually identified at the top of Figure 5 as a different array of dynamic spectra from the white noise initialization to the final result achieved, whereas visual inspection of the final result at the bottom confirms that the agent shown for each algorithm converged to the same attractor, representing a variant. Close examination of Figure 5 reveals that the different approaches pursued dissimilar trajectories upon performing the task of representing the same set of attractors. Notice in Figure 5 that, along the iterations, the trajectory resulting from the application of the AIS presents a very irregular pattern. It is heard as sounds differing a great deal

from one another. Also, the noisy contents disappeared very fast, in the first five iterations. SOM and PSO present similar features when visually examined. The noisy contents of the spectrum persists for more iterations and dies away more gradually. They both appear to be more homogeneous upon convergence, although PSO retained a little noisiness.

The abovementioned characteristics result in varied musical gestures emerging from the separate approaches applied to the same scenario. That is, a unique type of sound transformation is obtained with the application of each bio-inspired algorithm, summarized in Table 1. AIS searches the space in a jerky, lumpy way, resulting in discontinuous transitions between sounds, causing the impression of a bumpy transformation. SOM is the smoothest of all, gradually winding over the iterations in a snaky fashion until it converges. Finally, PSO is shaky and twitchy, with a spiraling gesture that resembles the swarming of insects around flowers. Table 1 also shows the total elapsed time each algorithm took to run on the same machine, for comparison purposes.

Also, it can be inferred from Figure 5 that, although different, the three algorithms successfully achieved their final objective of representing the attractors with different degrees of accuracy; resulting, in this case, in representations that correspond to variations of the original attractors. Although visually they seem to differ only in high frequency contents, that is, noisiness, perceptually they differ in sound qualities.

Moreover, this is true not only for the final result, but also for each iteration, where each agent occupies a distinct position in soundspace depending upon its interaction with its neighbors, the total number of agents searching the space, the strategy used, among other factors. That is, one can see that the emergence of stable, complex patterns (each individual agent representing each waveform) results from the populational strategy allied to the self-organizing features present in all algorithms. It is very important to notice that the proposal generates both form, represented by the trajectory, and matter, the individual agent in each iteration.

Table 1. Self-organizing features and sonic correspondence

	AIS	**SOM**	**PSO**
Trajectory	Jerky	Smooth	Shaky
Gesture	Discontinuous transitions	Gradual winding	Spiraling
Transformation	Bumpy	Snaky	Swarmy
Time	46.75 s	33.02 s	38.64 s

5 Conclusions

In this work we have proposed a populational approach to exploring soundspace in time domain with self-organization as generative and structuring paradigm by means of bio-inspired algorithms. Self-organizing systems feature the emergence of complex patterns through local interactions among simple individuals. Many physical and biological systems have been found to present patterns that appear to be self-organizing. A large number of bio-inspired algorithms have been proposed to solve a

wide range of problems in engineering as well as other areas, where traditional methods have failed. Here, we have shown the emergence of both musical form and matter by means of self-organizing temporal manipulation of sounds. In the system described here, the process of synthesis is integrated in the performance process so that the system can be used in musical performance. We have shown that the application of different bio-inspired paradigms leads to different paths followed by the agents in pursuing the attractors, which, in turn, results in different sound transformations and distinct variations of the attractors along the trajectory.

Future perspectives of this work include allowing the user to control the transformation in one or more musical dimensions (frequency, for example), reducing the self-organizing transformation to the remaining sound qualities. The adoption of moving attractors can greatly enhance the musical potential of the method, for the attractors themselves would be sound transformations. We also consider relevant to investigate other bio-inspired paradigms that could result in different transformations.

Acknowledgements

This work was fully developed under the supervision of Profs. Fernando Von Zuben and Jônatas Manzolli and supported by FAPESP (process 03/11122-8). The main author is presently supported by a grant from CAPES (process 4082-05-2) and advised by Prof. Xavier Rodet, who so kindly revised the text and made invaluable suggestions, greatly contributing to the improvement of the text.

References

1. Biles, J. A. (1994) GenJam: A Genetic Algorithm for Generating Jazz Solos, Proceedings of the 1994 International Computer Music Conference, (ICMC'94), pp. 131-137.
2. Blackwell, T. M., Bentley, P. Improvised Music with Swarms. Proceedings of IEEE Congress on Evolutionary Computation, 2002.
3. Blackwell, T. and Young, M. (2004) Swarm Granulator. In G. R. Raidl et al. (Eds): EvoWorkshops, Lecture Notes in Computer Science 3005, pp 399-408.
4. Bonabeau, E., Dorigo, M., Theraulaz, G. Swarm Intelligence: From natural to artificial syatems. New York: OxfordUniversity Press, 1999.
5. Burnet, F.M. The Clonal Selection Theory of Acquired Immunity, Cambridge University Press, 1959.
6. Burraston, D., Edmonds, E. A., Livingstone, D. and Miranda, E. (2004) Cellular Automata in MIDI based Computer Music. Proceedings of the International Computer Music Conference, pp. 71-78.
7. Caetano, M., Manzolli, J. and Von Zuben, F. J. (2005 a) Application of an Artificial Immune System in a Compositional Timbre Design Technique. In C. Jacob et al. (Eds): ICARIS 2005, Lecture Notes in Computer Science 3627, pp 389-403.
8. Caetano, M., Manzolli, J. and Von Zuben, F. J. (2005 b) Interactive Control of Evolution Applied to Sound Synthesis. in Markov, Z., Russel, I. (eds.) Proceedings of the 18th International Florida Artificial Intelligence Research Society (FLAIRS), Clearwater Beach, Florida, EUA, pp. 51-56.

9. Caetano, M., Manzolli, J. and Von Zuben, F. J. (2005 c) “Topological Self-Organizing Timbre Design Methodology Using a Kohonen’s Neural Network”. 10 Simpósio Brasileiro de Computação e Música, Belo Horizonte, Brazil.
10. Camazine, S., Deneubourg, J.-L., Franks, N. R., Sneyd, J., Theraulaz, G., Bonabeau, E. (2001) Self-Organization in Biological Systems. Princeton University Press.
11. Chen, C. J. and Miikkulainen, R. (2001) Creating Melodies with Evolving Recurrent Neural Networks, Proceedings of the International Joint Conference on Neural Networks (IJCNN-01), 2241-2246.
12. de Castro, L. N. & Timmis, J. I. Artificial Immune Systems: A New Computational Intelligence Approach, Springer-Verlag, London, 2002.
13. de Castro, L.N and Von Zuben, F. aiNET: An Artificial Immune Network for Data Analysis, in Data Mining: A Heuristic Approach. Abbas, H, Sarker, R and Newton, C (Eds). Idea Group Publishing, 2001.
14. Guéret, C., Monmarché, M., Slimane, M. Ants can play music. Fourth International Workshop on Ant Colony Optimization and Swarm Intelligence (ANTS 2004), Université Libre de Bruxelles, Belgique.
15. Kennedy, J. Particle swarms: Optimization based on sociocognition. In: Recent Developments in Biologically Inspired Computing, De Castro, L. Von Zuben, F. J. (Eds), Idea Group Publishing, ISBN:159140312X, 2004.
16. Kohonen, T. (2000) Self-Organizing Maps. Springer.
17. Miranda, E. R. Granular Synthesis of Sound by means of a Cellular Automaton. Leonardo, 28 (4), pp. 297-300, 1995.
18. Risset, J. C. Computer Study of Trumpet Tones. Murray Hill, N.J.: Bell Telephone Laboratories, 1966.
19. Smalley, D. 1990. Spectro-morphology and Structuring Processes. In The Language of Electroacoustic Music, 61-93.London: Macmillan.
20. Von Foerster, H. , “On Self-Organizing Systems and Their Environments.” In: Self-Organizing Systems, M. C. Yovits und S. Cameron (Hg.), Pergamon Press, London, pp. 31–50, 1960.
21. Wishart, T. “On Sonic Art”. Simon Emerson: Harwood Academic Publishers, ISBN 3-7186-5847-X, 1998.

An Evolutionary Approach to Computer-Aided Orchestration

Grégoire Carpentier[1], Damien Tardieu[1], Gérard Assayag[1], Xavier Rodet[1], and Emmanuel Saint-James[2]

[1] IRCAM-CNRS, UMR-STMS 9912, 1 place Igor Stravinsky, F-75004 Paris, France
[2] LIP6-CNRS, 8 rue du Capitaine Scott, F-75015 Paris, France

Abstract. In this paper we introduce an hybrid evolutionary algorithm for computer-aided orchestration. Our current approach to orchestration consists in replicating a target sound with a set of instruments sound samples. We show how the orchestration problem can be viewed as a multi-objective 0/1 knapsack problem, with additional constraints and a case-specific criteria formulation. Our search method hybridizes genetic search and local search, for both of which we define ad-hoc genetic and neighborhood operators. A simple modelling of sound combinations is used to create two new mutation operators for genetic search, while a preliminary clustering procedure allows for the computation of sound mixtures neighborhoods for the local search phase. We also show in which way user interaction might be introduced in the orchestration procedure itself, and how to lead the search according to the users choices.

1 Introduction

In the last decades Computer-Aided Composition (CAC) software have provoked a growing interest among contemporary music composers and have become a core element in the development of their works. Originally, motivation for the design of such tools was to provide composers with the ability to easily manipulate musical symbolic objects, such as notes, chords, melodies, polyphonies... Simultaneously, another main branch in computer music research concentrated its efforts on sound analysis, sound synthesis, and sound processing, leading to a finer comprehension of many aspects of the wide sound phenomenon, and among them, the timbre of musical instruments.

In the meantime, contemporary composers have slowly started - since the beginning of the 1970s - to move away from purely combinatorial aspects of musical structures, and have drawn their attention to the spectral properties of sound. This turning point in western orchestral music set up a new aesthetic direction that has been carried on by later and today's composers. Simultaneously, the parallel evolution of CAC made these pioneers and their successors dream of a composition tool that could cope with rich timbre information to help them in their orchestration tasks. Unfortunately, such a tool required that techniques from various fields of music research achieved a certain degree of

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 488–497, 2007.
© Springer-Verlag Berlin Heidelberg 2007

maturity. Today, the tremendous knowledge inherited from the analysis of instrumental sounds, the breakthrough in timbre research, the accessibility of large sound databases and the computer performance allow for the bridging of the gap between traditional CAC systems and the current potential of sound analysis and manipulation.

In a previous paper [1] we had presented a new tool for computer-aided orchestration, with which composers can specify a target sound and replicate it with a given, pre-determined orchestra. The development of this tool was driven by the wish to consider globally the complex mechanism of timbre perception and to allow the discovery of large, non-trivial solutions. Unfortunately, the NP-hardness of the problem discouraged us to expect orchestrations involving more than two or three instruments. In the present paper we introduce an hybrid, genetic/local-search algorithm designed to face the huge combinatorial problem arising in our orchestration procedure. This algorithm is mainly inspired by Jaszkiewicz's MOGLS [2], and has been significantly adapted to our specific case. The paper is organized as follows. Section 2 reports on previous work in the field of computer-aided orchestration. Section 3 recalls the main paradigms of our system and show why the orchestration procedure can be considered as a multi-objective knapsack problem. Our orchestration algorithm itself as well as specific genetic and neighborhood operators are then presented in Sect. 4. Last, conclusions and future work are discussed in Sect. 5.

2 Previous Works in Computer Orchestration

Computer-aided orchestration is a relatively new topic of interest in the computer music domain, and the literature in this field is somehow poor. In our precedent paper we had reviewed the three previous works that aim at designing orchestration tools. We briefly recall them here, for more details see [1].

Rose and Hetrik [3] propose a Singular Value Decomposition (SVD) -based algorithm that allows either the analysis of a given orchestration or the proposition of new orchestrations that approach a target sound. Another method proposed by Psenicka [4] addresses this problem by performing the search on instruments, not directly on sounds. In a Lisp-written program called SPORCH (SPectral ORCHestration), the author uses an iterative matching on spectral peaks to find a combination of instruments that best fit the target sound. The third system is proposed by Hummel [5]. The principle is similar to Psenicka's, except that it works on spectral envelopes rather than on spectral peaks. The program first computes the target's spectral envelope, then iteratively finds the best approximation.

All these methods present the significant advantage of requiring relatively low computation times. However, as they all rely on spectrum decomposition techniques (invoking either SVD or matching-pursuit methods), they implicitly consider a sound target replication procedure as filling a LIFO stack in which "bigger" elements are introduced first. Roughly speaking, these methods can been seen as kind of greedy algorithms which are known to achieve only approximations of the

best solutions in most problems. On the other hand, they fail in considering the timbre perception as a complex, multidimensional mechanism, as the optimization process is driven by a unique objective function. Our orchestration system and algorithm were designed to overtake these limitations.

3 Orchestration Viewed as a Multi-objective Knapsack Problem

3.1 Overview of Our Orchestration System

The general framework of our orchestration system is shown on Fig. 1. As presented in [1], one of the core concepts of our tool is the *target* object. This target is a set of audio and symbolic features that describe different aspect of the sound to be "reproduced" with an orchestra. These features may come either from the analysis of a pre-recorded sound, or from a compositional process.

The number of features is not fixed yet and will in all probability increase as research goes on. Currently we use little and static spectral data as audio description, and a set of pitches as symbolic features. This might seem somehow poor, however our purpose is not to build an exhaustive sound description, but rather to design a general framework which can be easily extended by adding new features when needed.

The target being defined, an orchestration engine uses an instrumental knowledge database (features database) created by the analysis and structuring of large sound sample databases, to suggest instruments notes combinations (orchestration proposals) that "realize" the target. More precisely, the procedure searches for combinations whose features best match the target's features. The orchestration proposals may afterwards be edited, transformed, or simulated.

3.2 The Multi-objective Knapsack Approach

Let E be the set of all sounds potentially produced by any individual instrument in a given orchestra, $P(E)$ the power set of E, T a target object, and $S(T)$ the set of elements of $P(E)$ that "sound" as close to the target as possible. Starting from an initial point K_0 of $P(E)$, our goal is modify K_0's elements in order to converge into $S(T)$. In other words, the elements of K_0 may be (at will) removed, substituted, or completed by other elements, provided that the total number of elements does not exceed the orchestra's size. As stated, the problem is extremely close to the Binary Knapsack Problem (KP-0/1), well known in operational research. The KP-0/1 is usually formulated as follows:

$$(\text{KP-0/1}) \begin{cases} \max z(x) = \sum_{i=1}^{n} p_i x_i \\ \\ \text{s.t. } x_i \in \{0;1\} \\ \quad \sum_{i=1}^{n} w_i x_i \leq C \end{cases} \tag{1}$$

where n is the size of the items set, w_i is the weight of item i, p_i the profit generated by inserting item i in the knapsack, and C is the total capacity of the

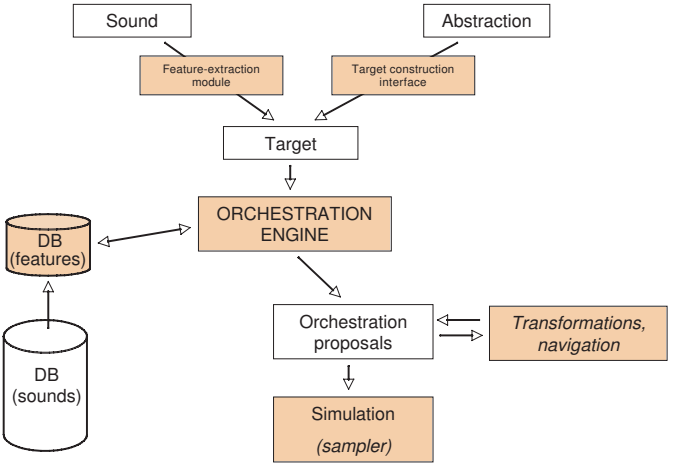


Fig. 1. General architecture of our orchestration tool

knapsack. In our orchestration context, the items are the sounds of the database, all the weights w_i are all equal to one and the capacity is the size of the orchestra. The definition of the profits is less straightforward and will be discussed in the next section.

As previously said, the target object is a set of features, and each feature is to be seen as a specific dimension of timbre. As we aim at capturing the timbre perception mechanism globally, all dimensions have to be considered jointly in the objective function. Unfortunately we cannot predict the relative contribution of each dimension, because we do not know *a priori* which target's characteristics the composer would like to reproduce, and most of the times neither does he (she).

The multi-objective approach is therefore mandatory. In multi-objective optimization the final output is not a *unique* solution but *a set of* efficient solutions, also called *Pareto-optimal solutions*. A solution is said *Pareto-optimal* when no other solution achieves better values on *every* criteria. For more details see for instance [2]. Formally, the Multi-Objective Knapsack Problem (MOKP-0/1) is stated as follows:

$$(\text{MOKP-0/1}) \begin{cases} \max z_k(x) = \sum_{i=1}^{n} p_i^k x_i \\ k = 1, ..., K \\ \text{s.t. } x_i \in \{0; 1\} \\ \quad \sum_{i=1}^{n} w_i x_i \leq C \end{cases} \tag{2}$$

where p_i^k is the profit of item i relative to the dimension (or criterion) k.

3.3 Constraints and Limitations

The MOKP-0/1 formulated in Eq. 2 cannot be applied directly to the orchestration problem. First, it is virtually impossible to define the profits without

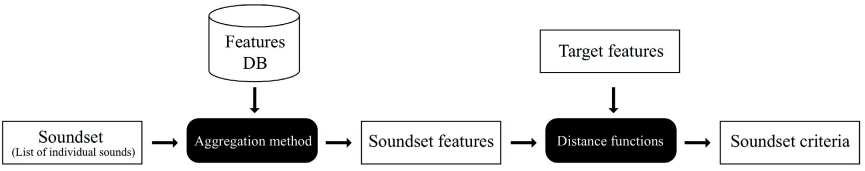


Fig. 2. Soundset criteria computation flowchart

knowing *in advance* all the elements in the combination. In other words, the profits are correlated; they are not anymore a function of a *single* index i, but of *all* indices $1, ..., n$. For instance, let x be an instrument sound sample, and K_1 and K_2 two sound mixtures defined as $K_1 = \{x\}$ and $K_2 = \{x, x\}$. Its is straigtforward that K_1 and K_2 have the same spectral features, as adding x to K_1 just increase its loudness. Figure 2 show how this problem can be overcome. Criteria are jointly computed each time a sound combination is created or changed. First, an *aggregation method* computes the combination features from the individual sounds' features. Then a set of *distance functions* compute the relative distances (along each timbre dimension) between the combination and the target. Distance are relative for homogeneity reasons. With such a formulation, the orchestration problem turns into a goal attainment problem, because we wish to minimize the distances to the target along each timbre direction, with an ideal value of zero for each criteria (when the goal is attained).

The other problem is related to orchestra's limitations. Re-using notations introduced in Sect. 3.2, a lot of elements of $P(E)$ are *not physically playable* by the orchestra, simply because a combination with two trombone sounds requires at least two trombone players, which is not necessary the case. We therefore introduce additional constraints for discarding non-feasible solutions. Let J be the number of different instruments in the orchestra and $I_j(i)$ a binary function, taking the value 1 if sound i is played by instrument j, 0 otherwise. The formulation of the Multi-Objective Orchestration Problem (MOOP) is now possible:

$$\text{(MOOP)} \begin{cases} \min z_k(x) = D_k(T, x_1, ..., x_n) & \\ k = 1, ..., K & \\ \text{s.t. } x_i \in \{0; 1\} & (a) \\ \quad \sum_{i=1}^{n} x_i \leq N & (b) \\ \quad \forall j \in \{1, J\}, \sum_{i=1}^{n} x_i I_j(i) \leq N_j & (c) \end{cases} \tag{3}$$

where N is the size of the orchestra, N_j the total number of instruments of type j, and $D_k(T, x_1, ..., x_n)$ the distance function between the target and a combination along dimension k.

4 An Hybrid, MOGLS-Inspired Algorithm

The operational research literature counts many efficient exact methods (such as Branch-and-Bound or dynamic programming) for solving the knapsack problem,

either uni- or bi-objective. For a complete review of these methods see [6]. However it is generally admitted that exact methods fail when the number of objectives is greater than two. Moreover, Branch-and-Bound techniques require the knowledge of items profits for the calculation of bounds, and are therfore inapplicable to the problem formulated in Eq. 3. Consequently, the use of heuristics methods is mandatory in our case.

Jaszkiewicz has proposed in [2] an efficient hybrid algorithm (called MOGLS) for multi-objective optimization, and has proved in [7] the superiority of MOGLS upon other methods for the MOKP-0/1. Basically, MOGLS is an evolutionary method that alternates genetic search and local search over the iterations, exploiting the fact that genetic and local search heuristics have different and complementary effects on populations of solutions. Here again, some preliminary concepts and operators need to be introduced before presenting our MOGLS-inspired algorithm in Sect. 4.5.

4.1 Preprocessing

Two preprocessing operations are performed before the orchestration procedure itself. First, the target is analyzed with a multi-f0 extraction method in order to discard a large set of candidates, as explained in [1]. Assume for instance the target's partial set can be explained by two pitches, C3 and Eb4. The candidate selection procedure will here keep only database sounds whose pitch belongs to C3 or Eb4 harmonic series, i.e. C3, C4, G4, C5, E5... and Eb4, Eb5, Bb5, Eb6, G6... respectively. We call "pools" these harmonic groups and "state" the harmonic rank in a given group. With such a terminology, a G4-pitched sound will belong to the C3 pool with state 3. Thanks to this concept we define ad-hoc genetic operators in section 4.3. After the multi-f0 extraction procedure a new database is created by filtering items by pitch and by instrument. The database items are sorted by increasing pitches, and within each pitch group, by increasing spectral centroid.

In a second step, the database items are clustered to form groups of sounds close to each other. The criterion used in the categorization phase is a euclidean distance on sounds contributions to the target's most important partials (for more details see [1]). In other words, this spectral distance is *target-specific*. We call "domain" of a given sound the set of all sounds belonging to the same cluster. This notion will be exploited in the local search procedure (see Sect. 4.4).

4.2 Modeling Sound Combinations

Sound combinations are modeled by a "soundset" object, containing two main slots: "elements" and "features". The elements field is made out of four vectors: the items indices in the sound database, the pools and states indices, and the instrument group vector. This last information is used to handle orchestra's

limitations (constraint (c) in Eq. 3). The pools vector is modified only when sounds are added or removed of the set (see Sect. 4.3).

Each time a sound combination is created or modified, features and criteria are computed as shown in Fig. 2. Then, a masking test procedure is invoked to "clean" the set by removing all non-perceptible components.

As suggested in [7] and [2] the combinations fitness is computed as a weighted aggregation of criteria with a weighted Tchebycheff function:

$$F(T, x_1, ..., x_n) = \max_k \lambda_k D_k(T, x_1, ..., x_n) \tag{4}$$

where $(\lambda_k)_{1 \leq k \leq K}$ are the aggregation weights, randomly drawn on each iteration.

4.3 Genetic Operators

We use the conventional binary string representation for the genetic encoding, and the crossover operator is the classic 1-point crossover. Mutation, however, is defined differently than the traditional bit-flip. In fact, we use three different kinds of mutations:

1. Horizontal shift: The sound is replaced by another one with same pool and state, but the instrument, the playing style and the dynamics may differ. The substitute sound is chosen to have a spectral centroid value as close as possible to the original.
2. Vertical Shift: The sound is replaced by another one with same instrument and pool, but the state may differ. The probability of an up shift is lower than the probability of a down shift.
3. Addition or Deletion.

An illustration of all genetic operators is show on Fig. 3.

4.4 Adaptive Search for Local Exploration

Local search is used to search for better solution within the neighborhood of sound mixtures. The method is inspired by adaptive local search introduced by Codognet and al. [8]. Here again the objective function is a weighted Tchebycheff distance with random weights. The variables are the combination's items and the domains are their associated classes as built during the clustering phase (see Sect. 4.1). As we cannot define a marginal cost function for each sound in the set, the currently selected variable is the non-taboo item with the highest loudness value.

4.5 Orchestration Algorithm

MOGLS for MOOP. Our MOGLS-like orchestration method uses a population of solutions, each of them modeling a sound combination as explained in

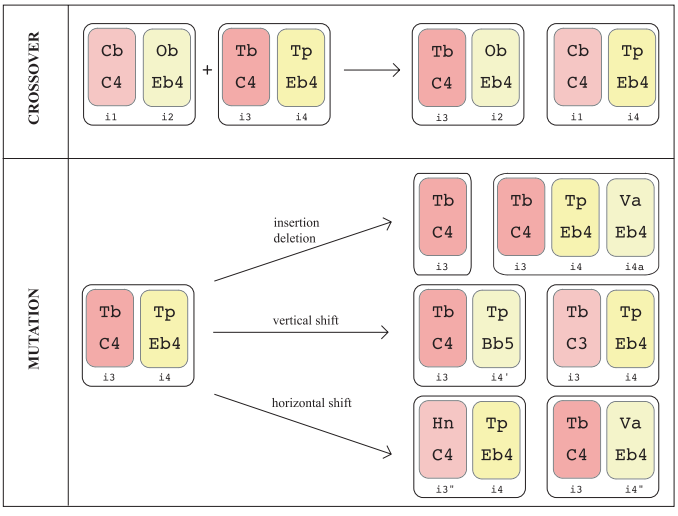


Fig. 3. Genetic operators for an orchestration algorithm

Sect. 4.2. At each iteration, a set of weights is drawn randomly and fitness is computed. Then, individuals are chosen to fill the mating pool with a binary-tournament selection method. Genetic operators then apply and offsprings are inserted in the population if the corresponding mixtures are playable by the orchestra. Afterwards, a new set of weights is drawn and another set of N best individuals are selected on the basis of the new fitness values for local-search improvement. As recommended by Ishibushi and al. in [9], The genetic part and the local search part have been kept separated. Furthermore, the mating pool size and the number of local search iterations have been chosen to allow equal computations times to both phases, as also suggested by the authors in [9].

User Interaction. As explained in [1], user interaction was thought as a fundamental paradigm in the design of our system. Interaction is introduced in the orchestration algorithm itself in the following way: When the algorithm reaches the maximum number of iterations, the user is asked to choose the solution of the Pareto set he (she) finds the closest of the target. New aggregation weights are then computed in ordered to rank the chosen solution first on the fitness scale. These weights reflect the user preferences and are calculated as in Eq. 5:

$$\lambda_k = \frac{D_k^{-1}(T, x_1, ..., x_n)}{\sum_k D_k^{-1}(T, x_1, ..., x_n)} \tag{5}$$

We then explore the direction of the search space implicitly suggested by the user's choice. The algorithm remains the same, but the weights have know fixed values, as calculated with Eq. 5. The overall orchestration procedure schema comes hereafter.

Orchestration Procedure

1. Build a target object and perform mutli-f0 analysis on target.
2. Build new sound database, by filtering on pools and instruments.
3. Sort items and compute sound clusters.
4. Build initial combinations population.
5. Until a stopping criterion is met, do:
   - Draw random weights and compute fitness with Eq. 4.
   - Fill mating pool with a binary tournament selection scheme.
   - Apply crossover and mutation operators.
   - Draw new random weights and re-compute fitness.
   - Select N best individuals and improve them with local search.
6. Ask the user to choose *one best solution* in the current Pareto set.
7. Compute the user's weights with Eq. 5.
8. Goto step 5 with the new fixed weights. Repeat procedure at will.

4.6 Results

Evaluation methods are difficult to design in our case because there is no measure of performance adapted to the orchestration problem. Traditionally performances of multi-objective optimization methods are based on the size, shape, density, or homogeneity of the Pareto set, or the distance between the theoretic Pareto set and its approximation, when the former is known. In the orchestration problem the theoretic Pareto is only known when the target is *exactly playable* by the orchestra, for instance a mixture of the instrument database sounds. In that specific case the theoretic Pareto set is reduced to the target itself. Otherwise, Pareto sets obtained by our algorithm are difficult to score. In all probability the evaluation procedure will mostly depend on composers expectations.

Early experiments with our system however gave encouraging results. Examples of pre-recorded sound target orchestrations are available on the following web page:

http://recherche.ircam.fr/equipes/analyse-synthese/dtardieu/exemple_orchestration.html

5 Conclusions and Future Work

In this paper we have exposed how an orchestration task could be viewed as a variant of the Multi-Objective Knapsack Problem (MOKP-0/1). A refinement of criteria computation and a set of extra constraints have been introduced to define the Multi-Objective Orchestration Problem (MOOP). An evolutionary algorithm inspired of Jaszkiewiecz's MOGLS have then been proposed to address the MOOP. This algorithm is a hybrid method where genetic search and local search alternate. We have also explained how user interaction could be introduced in the search procedure itself, and how the user's choice among the current solutions could help in guessing its preferences.

Future research will focus on the design of evaluation measures and procedures for our method. Alternative approaches to MOGLS for multi-objective search might then be tested. In the meantime, Gaussian Mixture Models (GMM) -based

instruments models trained on large instruments sample databases should help on one hand in increasing the power of generalization of our system, on the other hand in defining appropriate combinations neighborhoods. This should significantly ease both genetic and local search procedures.

Acknowledgments

Once again the authors would like to deeply thank the composers Yan Maresz and Joshua Fineberg for their involvement in the project. Their everyday support is always a great source of motivation for future development and research.

References

1. Carpentier, G., Tardieu, D., Assayag, G., Rodex, X. and Saint-James,E.: Imitative and Generative Orchestrations Using Pre-analyzed Sound Databases. Proc. of Sound and Music Computing conference, Marseille, France (2006) 115–122 http://mediatheque.ircam.fr/articles/textes/Carpentier06a/
2. Jaszkiewicz, A.: Genetic Local Search for Multi-Objective Combinatorial Optimization. European Journal of Operational Research (2002)
3. Rose, F. and Hetrick, J.: Spectral Analysis as a Ressource for Contemporary Orchestration Technique. Proc. of Conference on Interdisciplinary Musicology (2005)
4. Psenicka, D.: SPORCH: An Algorithm for Orchestration Based on Spectral Analyses of Recorded Sounds. Proc. of International Computer Music Conference (2003)
5. Hummel, T.: Simulation of Human Voice Timbre by Orchestration of Acoustic Music Instruments. Proc. of International Computer Music Conference (2005)
6. Martello, S. and Toth, P.: Knapsack problems : Algorithms and computer implementations. John Wiley & Sons, Chichester (1990)
7. Jaszkiewicz, A.: Comparison of local search-based metaheuristics on the multiple objective knapsack problem. Foundations of Computing and Design Sciences 26 (2001) 99–120
8. Codognet, P., Diaz, D., and Truchet C.: The Adaptive Search Method for Constraint Solving and its Application to Musical CSPs. 1st International Workshop on Heuristics (2002)
9. Ishibuchi, H., Yoshida, T., and Murata., T.: Balance between Genetic Search and Local Search in Hybrid Evolutionary MultiCriterion Optimization Algorithms (2002)

Evolution of Animated Photomosaics

Vic Ciesielski[1], Marsha Berry[2], Karen Trist[2], and Daryl D'Souza[1]

[1] School of Computer Science and Information Technology
RMIT University, GPO Box 2476V, Melbourne Vic 3001, Australia
vc@cs.rmit.edu.au
http://www.cs.rmit.edu.au/∼vc
[2] School of Creative Media, RMIT University

Abstract. Photomosaics are images composed of smaller images (tiles). Viewed close up the details of the tiles are evident and the big picture is lost. Viewed from a distance the detail of the tiles is lost and the big picture is evident. We show how photomosaics can be generated by evolutionary search and how animations can be created by using the best individuals in a generation as frames of a movie. The animations can generate engaging visual effects such as gradually 'materialising' a face from a random arrangement of tiles on a screen. Our animations explore self representation through dynamic reinterpretations of mosaic and iconography traditions using target images and tiles that we created.

1 Introduction

Traditional mosaics are images that are composed using a technique of using many pieces of glass, stone or ceramic tiles of similar shapes and sizes to make up a larger image. Some famous examples may be found at Pompeii, Ravenna and Herculaneum in Italy and Eastern Orthodox churches in Russia and Greece. Modern examples include works by Gaudi and Mexican artists Diego Rivera and Juan O'Gorman [1].

Photomosaics are a digital age refinement of traditional mosaics. In a photomosaic the tiles are small images which are interesting in their own right and their arrangement makes up another large image. Photomosaics are visually interesting because they are different when viewed from close up and from afar. When viewed from close up the details of the individual tiles interest the eye. When the viewed from afar, the detail in the tiles is no longer visible but a new image is apparent. In one poignant example the tiles are photos of soldiers killed in Iraq, but when viewed from afar the image is of President George Bush [2]. In some of our work we have used images of the authors as targets (figure 4d) and miniature images of the authors as tiles (figure 2, bottom row).

Traditionally, photomosaics are a static art form, and need to be viewed on paper images that have been printed in high resolution so that the detail of the individual tiles are evident to the viewer. A lot of this fine detail is lost on a

[1] Examples of evolved photomosaics and animations can be found at www.cs.rmit.edu.au/∼vc/evolved-images/mosaics

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 498–507, 2007.
© Springer-Verlag Berlin Heidelberg 2007

computer screen. We have been experimenting with genetic search to generate photomosaics. We discovered that animations of the search process were visually appealing to us. The original image was a random screen of tiles and by treating the best of each generation as a frame of a movie we obtained some striking, tile-shuffling effects as the subject of the image gradually "materialised" from the initial random screen.

The aim of this paper is to describe an approach to the generation of photomosaics using genetic algorithms, to show how visually interesting animations can be generated by visualizations of the search process and to reveal process behind an artwork that constitutes our notion of self representation through evolution to generate dynamic expressions.

2 Related Work

Historically, in the context of digital image processing, photomosaic work (and the term *photomosaic*) is attributed to the stitching together of adjacent pictures into a compound photograph. The idea of using a computer to decompose a digital image into building blocks has its origins in *DominoPix*, a computer graphics system attributed to Ken Knowlton [3]. The system may be used to construct pictures out of complete sets of dominoes. Photomosaics constructed from smaller, computer-generated tiles that match the overall image were invented by Robert Silvers [4,5]. Silvers conceived of the idea of dividing a source (or target) image into regions and comparing them with source image portions, to determine the best available matching source image. Silvers's company, Runaway Technology, produces photomosaic stills and animations. His photomosaic portrait of Bill Gates [5] was exhibited in the National Portrait Gallery in London thus establishing photomosaics as art.

Research into automatic generation of photomosaics is increasingly attracting attention. Indeed, the problem of transforming a set of image tiles into a "best fit" mosaic has evoked some interesting questions. What might the impact be if tiles are not of equal size, but instead may vary in size and shape (as well as other attributes)? And, what of the impact if tiles may be rotated, mutated or constrained in terms of their position? Should tiles be manually selected or generated, a priori, or should they be retrieved from very large image databases, a process that may be repeated at each iteration of the transformation process, from the formative tile set to the final, target image? We briefly survey research into automatic photomosaic generation.

Quantitative methods for generating photomosaics were first published by Tran [6]. They proposed two measures to quantify the distance between corresponding photomosaic image tiles and target image blocks. These measures were evaluated according to a range of criteria. To select tiles, one algorithm computes distance as the sum of the absolute difference between red, blue and green pixel component scores. In the other algorithm string alignment scores are computed between rows of image blocks and tiles. Hausner [7] devised an

algorithm to maximise the square tile coverage (and minimise "grout") in generating decorative mosaics. They use a method known as the centroidal Voronoi diagram [8,9,10], which allows for simultaneous optimisation of tile positions. Tiles may be rotated to maximise coverage. Kim and Pellacini [11] use a mosaicing technique known as Jigsaw Image Mosaic, where tiles of arbitrary shapes are used to compose the final image. They use a framework in which they seek to minimise a "mosaicing energy function", to select the optimal tile configuration. The energy function penalises configurations that do not maintain the colour of the target image. Their work was inspired by the work of the Renaissance Italian painter, Guiseppe Arcimboldo [12], who invented a style of painting that used clusters of images of fruit, vegetables and other materials, to paint human faces. Di Blasi [13] developed improvements to the algorithms of Hausner [7] and Kim and Pellacini [11], but could not evaluate such improvements against Silvers' algorithm, because the latter is protected by a patent. In another investigation, Li et al [14] use a content-based image retrieval (CBIR) of 50,000 images to support the generation of a series of destination artistic mosaics, from an arbitrary, original image.

Animations of photomosaics produce interesting effects where the final picture is revealed through an iterative process of gradual disclosure, via generation of improved best-fit frames at each iteration. These offer much to the artist and are already used by scientific data imaging [15]. Klein et al. [16] develop a distance measure to assess based on average colour and three-dimensional wavelet decomposition signatures in the colour space. Smith et al. [17] explore the generation of *temporally coherent* mosaics–individual frames are chosen to present a coherent sequence of evolving frames, in time.

3 The Evolutionary Algorithm

Our approach requires the artist to provide a target image and a fixed set of tile images. We view the canvas as a two dimensional grid and the generation of a photomosaic as an optimization problem in which the task is to find a selection and arrangement of tiles in the grid that is most similar to the target. We use a rearrangement based genetic algorithm which has been implemented using sga-c [18] to perform the search.

3.1 Representation

A potential solution is an arrangement of tiles in a two dimensional grid that is the same size as the target image. If the target is of size $N \times M$ and the tiles are of size $n \times m$, we require that n divides N and m divides M. $N/n = C$ tiles are required to fill a row of a potential solution and $M/m = R$ tiles to fill a column. A potential solution is thus an $R \times C$ array of integers in which each integer is an index into the tile set provided by the artist. If t tiles are provided then each integer is between 1 and t. A chromosome is this matrix represented as a single vector in row major order.

Crossover can only occur at points that are integer boundaries. We have used 1 point crossover, with two children being created from two parents. The crossover operator is applied with probability *Crossover Rate*. If crossover does not occur, the parents are simply copied to the children. Mutation is implemented by picking a random position in the array and replacing it with random integer between 1 and t. *Mutation Rate* is interpreted as the probability of changing a particular allele.

3.2 Fitness Evaluation

For grey level images, fitness is evaluated by summing the differences in pixel grey levels between the candidate solution and the target (equation 1). For colour images the sum is taken over the red, green and blue components.

$$\sum_{i=1}^{N}\sum_{j=1}^{M}|target(i,j) - individual(i,j)| \tag{1}$$

4 Technical Experiments

Table 1 shows the basic configuration used in the experimental work. The parameter values were determined through preliminary experimentation. Figure 1 shows the convergence behaviour for a number of choices of parameter values for one set of tiles and one target. It is difficult to choose an optimal set of parameter values since the convergence behaviour changes for different tile sets and targets. Overall, mutation rate was the most critical parameter with 0.0001 being a good value. In generating the animations described in this paper, the chromosomes contained about 2,500 alleles and this mutation rate corresponds to 2-3 mutations per evolved individual. Runs with mutation rates greater or smaller than 0.0001 generally required more evaluations for convergence. Runs with this mutation rate generally found the best solution by 20,000 generations.

We have designed and implemented this genetic algorithm primarily to provide an easily implemented platform for experimentation by the artists. It is possible

Table 1. Genetic Algorithm Configuration

Parameter	Value
Population Size	120
Crossover Rate (C in figure 1)	0.7
Mutation Rate (M in figure 1)	0.0001
Elitism Rate (E in figure 1)	0.1
Max Generations	50,000
Selection	Proportional to fitness
Termination	100,000 generations
Replacement	Generational replacement

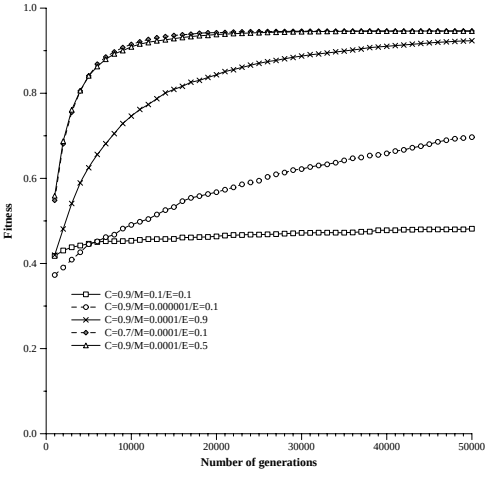


Fig. 1. Convergence behaviour for a selection of parameter values, averages of 5 runs

that different design and implementation choices will lead to animations with different dynamics. We leave this investigation for further work.

5 Construction of the Animations

Each time a new best image was found in an evolutionary run it was rendered and written as an image file. These images form the individual frames of a movie. Some experimentation was needed to determine the best way of constructing an animation. Using every frame often led to animations that were too boring because things changed too slowly. Taking every kth, k between 1 and 10 worked well in most cases, with choice of k being very much dependent on the target image and tile set. In some situations using a different k at different stages of the animation could lead to enhanced effects, for example, if the target image is a face there is a period in which it is clear that something is emerging but it is not clear what. The curiosity and anticipation of the viewer could be enhanced by using more frames at this stage and drawing out the recognition process.

By copying the frames that make up an animation, reversing the order and then adding them to the original animation we have achieved a very engaging effect. The screen starts off a with a random configuration of tiles, as in figure 3a, a recognizable images 'materializes' and then 'dematerializes' back to random. Such animations are even more compelling when played in a loop.

Unfortunately there is a large variability in the available tools for constructing movie files from individual frames and for playing the movies. We found that the same animation file was rendered differently by different viewers. Very pleasing artistic effects achieved with one viewer were lost with another viewer. Renderings by some viewers were atrocious. The speed of the computer processor also affected the rendering. We found that constructing MPEG files with PPMTOMPEG and

viewing them with the VLC viewer on a PC or the MPLAYER viewer on a Solaris environment generally worked well.

6 Artistic Experiments

There are two major challenges for the artist in using our system. The first is to conceptualize and produce interesting targets and tile sets. The second is to deal with the size and resolution limitations of a computer screen. Also, aesthetic criteria for what makes a good static mosaic rendered on a large canvas are different to those for a dynamic image rendered on a computer screen.

We decided that the human face was the most suitable subject given the technical limitations. The subject matter of the human face has a long history in art, both oriental and occidental. The human face is readily recognisable, no matter how distorted or how little information is on offer. McLeod [19] notes that a circle with two dots is interpreted as a face. The eyes are the key to the face. Babies learn to recognise the human face and differentiate between their mother's face and strangers early on. We worked in grey scale because colour is a distraction from the pure form of the face and working with limited tile sets we lacked the ability to control nuances in colour value. The place of randomness and chance effects were important in two ways. First, randomness is a key to evolution and secondly, randomness is a concept that has occupied the attention of key artists of the twentieth century.

Dada [20, p534] defined itself as being framed by notions of randomness and chance effects, for example Marcel Duchamp. Cubism [21, p730] was characterized by the breaking up and reconfiguring subject matter into an abstraction. With the emergence of new and digital media, remix culture where sound loops, video clips and images are re-used and recombined to produce new works has become a hallmark of postmodernism. Manovich, in his essay, Flash Generation [22], analyses this phenomenon and concludes that the challenge is to integrate the modernist paradigm (with its belief in science, industrialization and efficiency) with the postmodern paradigm (with its belief in skepticism, the marginal, the complex and the opaque) through remixing using programming and computing technology in order to develop new aesthetics.

We needed to reconcile the tension between the total number of tiles and tile size per image. If we had the total number of tiles that would allow us to select

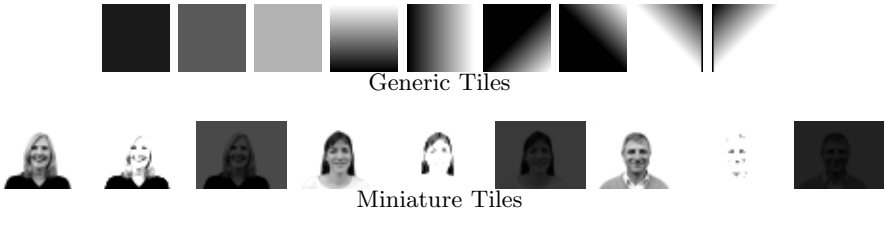
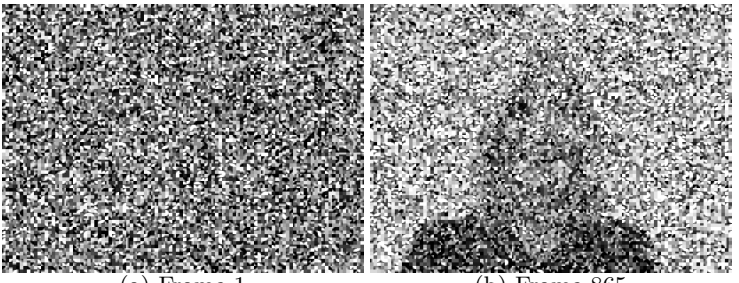
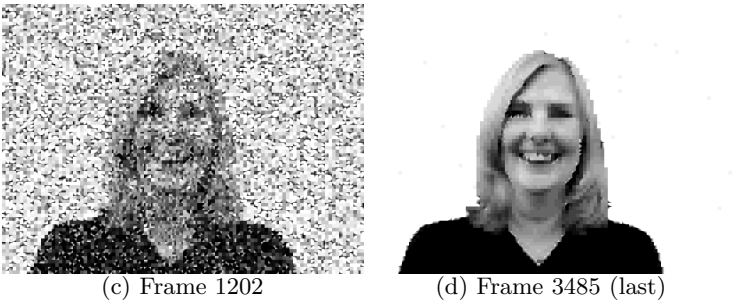


Fig. 2. A selection of the two tile sets used in the animations

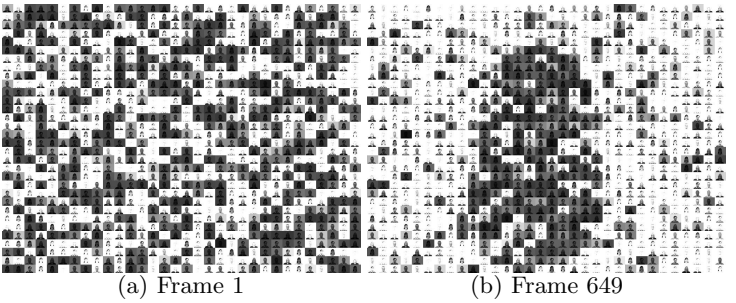


(a) Frame 1 (b) Frame 865

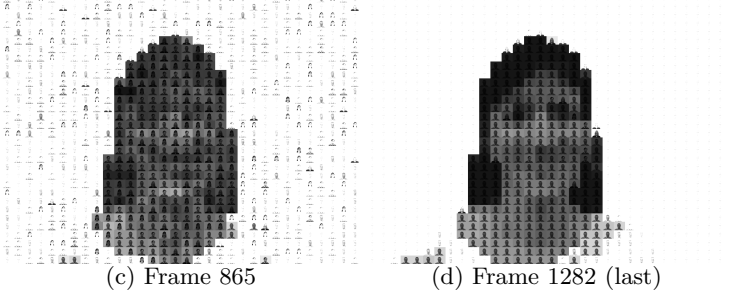


(c) Frame 1202 (d) Frame 3485 (last)

Fig. 3. Frames from animation, generic tiles



(a) Frame 1 (b) Frame 649



(c) Frame 865 (d) Frame 1282 (last)

Fig. 4. Frames from animation, miniature tiles

and subtly render fine details and colour relations we lost detail in the miniature image of the tiles. This was due to the actual size we could make the image, in turn based on the size of a screen. One can scroll around a still image but with an animation it is important that the viewer can see the complete image throughout the animation sequence.

Our artistic intention was to create iconic images of the face drawn from the Byzantine tradition [23,20, p157]. Icons dealing with religious themes have a history extending to Byzantium in 332AD [24,25] The figures are non-naturalistic and appear to float in the background landscape. Perspectives are distorted and flattened. The key features of faces are emphasized [26].

An iconic image of the human face is one that focuses on the cardinal points; eyes and mouth rather than a recognisable image of a face. We were inspired by the French conceptual artist Christian Boltanski's iconic memento mori *Altar to the Lycée Chases* [27,28, pp. 372-3] and the Shroud of Turin [29].

Figures 3 and 4 are snapshots from some of the animations we have created. To generate the target images we took portraits of three of the authors using a mobile phone with a 1.3 megapixel camera. The images were low resolution and deliberately shot in light that would maximise contrast and highlight the cardinal points of the face. We chose a plain background for this reason as well.

We have experimented with two different tile sets: (1) generic tiles and (2) miniature tiles. The generic tiles are essentially a set of grey level gradients rendered at different average intensities. Some examples are given in the top row of figure 2. This tile set was used in the animation shown in figure 3. The miniature tiles were constructed by reducing a target image to the required tile size and rendering it at a range of grey levels, both lighter and darker than the original. Some examples are shown in the bottom row of figure 2. This tile set was used in the animation shown in figure 4.

The animation sequences shown in figures 3 and 4 are part of our work "Constantly Becoming Other". In this animation, inspired by icons and the Shroud of Turin, we used remix culture to evolve a self referential work showing our faces dynamically dissolving into each other in a dynamic way. This work shows a 'materialize' and 'dematerialize' sequence of each author/artist in turn using the generic tiles followed by a similar sequence using the miniature tiles.

7 Conclusions

In this work our objective was to explore evolutionary search for the purpose of constructing photomosaics that expressed our concept and our chosen aesthetic. We have developed a system that generates photomosaics from a set of tiles and a target image. We discovered that animating the search by creating a movie using the best-of-generation individual gave visually engaging effects when the movie was replayed. The system provided creative opportunities for artists. We carefully selected target images and tiles based on a number of artistic traditions and created animations that explored self representation through dynamic expressions. The evolutionary algorithms provided us with a medium through

which we could express our concept about constantly becoming other literally and metaphorically. Evolutionary algorithms provided the opportunity to experiment with remix culture to provide dynamic reinterpretations of old forms like mosaics. We believe that the animations are sufficiently captivating to be exhibited at curated events.

We found that the generic tiles give better animations, but the animations are not as interesting from close up. Also, the fitness function was not totally satisfactory. It produced too many areas where the tiles were the same. However, this just makes the final photomosaic less interesting as a static image. The animation maintains interest because it is always changing.

We think that there are further opportunities for evolving interesting art by extending the work to colour images, using tiles of different sizes, evolving the tiles using a co-evolutionary approach and exploring different genetic algorithm design configurations.

Acknowledgments. We thank Austin Wood for coding the original mosaic program, Irwan Hendra for subsequent modifications, Emil Mikulic and Peter Papagiannopoulos for technical assistance.

References

1. Lewis, B., McGuire, L.: Making Mosaics. Drake Publishers, New York (1973)
2. Anon: The war president. http://www.michaelmoore.com/_media/images/special/the_war_president_hires.jpg (Visited 06-Oct-06)
3. Knowlton, K.: DominoPix, US Patent No. 4,398,890, Representation of Designs. http://www.knowltonmosaics.com/, http://www.metron.com/DominoPix/ (1983) Visited 02-Nov-06.
4. Silvers, R., Hawley, M.: Photomosaics. Henry Holt and Company, Inc., New York, New York (1997)
5. Runaway Technology: Company web site. http://www.photomosaic.com/rt /highlights.htm (Visited 06-Oct-06)
6. Tran, N.: Generating photomosaics: an empirical study. In Bryant, B., Lamont, G., Haddad, H., Carroll, J., eds.: Proceedings of the 1999 Symposium on Applied Computing, San Antonio, Texas (1999) 105–109
7. Hausner, A.: Simulating decorative mosaics. In Fiume, E., ed.: Proceedings of the ACM-SIGGRAPH International Conference on Research and Development in Graphics and Interactive Techniques, New York, CA (2001) 573–580
8. Okabe, A., Boots, B., Sugihara, K., Chiu, S.N.: Spatial Tessellations—Concepts and Applications of Voronoi Diagrams. 2 edn. John Wiley (2000)
9. Aurenhammer, F.: Voronoi diagrams—a survey of a fundamental geometric data structure. ACM Computing Surveys **23**(3) (1991) 345–405
10. Du, Q., Faber, V., Gunzburger, M.: Centroidal Voronoi tessellations: Applications and algorithms. SIAM Review **41** (1999) 637–676
11. Kim, J., Pellacini, F.: Jigsaw image mosaics. In: Proceedings of the ACM-SIGGRAPH International Conference on Research and Development in Graphics and Interactive Techniques, San Antonio, Texas (2002) 657–663
12. Strand, C.: Hello, Fruit Face!: The Paintings of Giuseppe Arcimboldo. Prestel (1999)

13. Blasi, G.D.: Fast techniques for Non-Photorealistic Rendering. PhD thesis, University of Catania (2006)
14. Li, X., Yuan, Y.: Artistic mosaic generation. In: Proceedings of the 3rd International Conference on Image and Graphics, Hong Kong (2004) 528–531
15. NASA: Mapping the Amazon: Mosaic tiles animation. http://svs.gsfc.nasa.gov/vis/a000000/a002400/a002405/ (Visited 06-Oct-06)
16. Klein, A., Grant, T., Finkelstein, A., Cohen, M.: Video mosaics. In: Proceedings of the 2nd International symposium on Non-photorealistic animation and rendering, Annency, France (2002) 21–28
17. Smith, K., Liu, Y., Klein, A.: Animosaics. In: Proceedings of the ACM-SIGGRAPH International Conference on Research and Development in Graphics and Interactive Techniques, Los Angeles, CA (2005) 201–208
18. Smith, R.E., Goldberg, D.E., Earickson, J.A.: SGA-C: A C-language implementation of a simple genetic algorithm (1991) http://citeseer.ist.psu.edu/341381.html.
19. McLeod, S.: Understanding Comics. Kitchen Sink Press and Paradox Press (1993)
20. Janson, H.: A History of Art: A Survey of the Visual Arts from the Dawn of History to the Present Day. Thames and Hudson, London (1972)
21. Honour, H., Fleming, J.: A World History of Art. 4th edn. Laurence King Publishing, London (1995)
22. Manovich, L.: Generation flash. In: Multimedia and the Interactive Spectator. An international workshop, University of Maastricht (2002) http://www.fdcw.unimaas.nl/is/generationflash.htm, Visited 02-Nov-2006.
23. Weitzmann, K.: The Icon, Holy Images: Sixth to Fourteenth Century. Chatto and Windus, London (1978)
24. L'Orange, H., Nordhagen, P.: Mosaics: From Antiquity to the Early Middle Ages. Methuen and Co, London (1966)
25. Cormack, R.: Painting the Soul: Icons, Death Masks and Shrouds. Reaktion Books, London (1997)
26. Mango, C.: The Art of the Byzantine Empire, 3121453: Sources and Documents. Prentice-Hall, Englewood Cliffs, NJ (1972)
27. Boltanski, C.: Altar to the Lycée Chases. http://www.moca.org/museum/pc_artwork_detail.php?acsnum=89.28&keywords=boltanski&x=0&y=0& (Visited 06-Oct-06)
28. Greenough, S.: On the Art of Fixing a Shadow: One hundred and fifty years of Photography. National Gallery of Art, Art Institute of Chicago (1989)
29. Gove, H.: Relic, icon, or hoax? Carbon dating the Turin shroud. Institute of Physics Pub, Philadelphia (1996)

Environments for Sonic Ecologies

Tom Davis and Pedro Rebelo

Sonic Arts Research Centre, Queens University Belfast,
University Road, Belfast, BT7 1NN
{tdavis01, P.Rebelo}@qub.co.uk

Abstract. This paper outlines a current lack of consideration for the environmental context of Evolutionary Algorithms used for the generation of music. We attempt to readdress this balance by outlining the benefits of developing strong coupling strategies between agent and environment. It goes on to discuss the relationship between artistic process and the viewer and suggests a placement of the viewer and agent in a shared environmental context to facilitate understanding of the artistic process and a feeling of participation in the work. The paper then goes on to outline the installation 'Excuse Me and how it attempts to achieve a level of Sonic Ecology through the use of a shared environmental context.

Keywords: Evolutionary Music, Ecology, Environment, Installation, Agent, Artwork.

1 Introduction

Within the field of Artificial Intelligence there has been a relatively recent embracement of Ecological Theory [1], with a large body [19] of research now centering on the importance of an agent's embodiment within an environment and the consequential contribution to the growth and development of a situated definition of cognition. This ecological approach moves away from the classical idea of a dualism of mind and body [21] towards a philosophy where the interaction between the agent and its environment constitutes a reciprocal relationship which is defined by the agents physical embodiment within an environment and its ecological relationship to it.

Despite a long tradition of site specific composition in music, with examples as far back as the Baroque (Gabrielli, Bach [24]); developments of concepts of space and situatedness in composition (Mozart, Serenade in D for 4 Orchestras (K286 1777), Stockhausen, Gruppen for 3 Orchestras 1955,[24]); the later obsession with multi-channel speaker transmission by electroacoustic composers [21] and the embracement of a concept of the environment by groups such as 'The World Soundscape Project', founded by Schafer [23] and publications such as Truax's 'Acoustic Ecology' [26], it is only recently that theorists and artists such as Whitelaw [28] and McCormack [16][17] have started calling for a similar level of environmental consideration in systems for the generation of sound/music that utilise models from the field of Artificial Life.

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 508–516, 2007.
© Springer-Verlag Berlin Heidelberg 2007

2 The Un(der)-Modeled Environment

Whitelaw turns to the notion of 'system stories', 'a translation or narration of the processual structures, ontology, entities and relations in a software system' [28], as a way of critiquing generative art works built around a system code. He argues that it is these 'system stories', which must be read as the core of the work. Whitelaw draws a parallel between the 'system stories' of generative artworks and the more complex social and cultural systems we live in and notes a missed opportunity for the creation of 'system stories that are more sophisticated, critical or experimental'. It is perhaps this gap in sophistication of 'system stories' of generative art that creates difficulties for viewers/listeners to form meaningful relationships to such model worlds, which Whitelaw dubs 'pure machines' and 'clockwork constellations'. Whitelaw uses emergent swarms as an example of a 'system story' that could be considered to have a 'weak' narrative.

> 'But consider the subject or agent modeled here, if thats the story we want to tell: a clone in a crowd, unchanging, with no traction on the space it inhabits, existing in an ongoing, perpetual present. If these systems provide images of contemporary society then they are, at best, nave and utopian' [28].

The environment as modeled in Whitelaw's example could be considered to be amnesic [14]. It has no memory nor history of interaction within the systems programmed environmental context. As Jon McCormack stated '[o]ne common oversight made by those trying to evolve creative systems is proper consideration of environment' [17]. Many systems taken from the field of Artificial Life used for the generation of sound/music are based on models that utilise 'non-reductionist' techniques designed to exhibit emergent structures, which almost always consist of a number of agents existing within a shared virtual environment. The strength of environmental coupling between agent and environment can vary from weak to strong as programmers determine not only the nature and strength of the interactions between the individual agents but also the quantity and form of any environmental reciprocity. Such environmental interactions tend to be striped back to a bare minimum to allow for an easier understanding of the underlying process but as Whitelaw suggests such a level of abstraction is perhaps missing an opportunity for creating something ultimately more meaningful?

We propose that a system that models an environment that is persistent [14], (as opposed to amnesic), dynamic (rather than static), an environment that takes on more than just the role of message transport, a system that treats the environment itself as a 'first-class' entity [14], has a much better chance of developing a 'system story' sophisticated enough to be understood to have a relationship to the systems of the 'outside' world. Some recent environmental models [7][8][25] have included environmental factors such as food sources, disease and even evolving critics and this deeper relationship between environment and agent seems to give an 'aesthetic and generative payoff' [28]. Yet a truly reciprocal relationship between agent and environment seems to be lacking. We propose these types of considerations are especially important for the implementation of Artificial Life

based systems for the generation of sound/music. As there are strong parallels between the generation of complex 'system stories' and the social structure of music making found in reaction to a dynamic and persistent environmental and cultural context, and in the varied performance and compositional practices.

We therefore suggest that more though needs to be given to the process of embodiment of the agents within their environment; the level of environmental coupling that is afforded to them and the process by which the environment mediates their interactions. We propose that a deeper relationship between agent and environment, one in which there is a level of mutual exchange between the two. Where environmental constraints and mediation can lead to unexpected outcomes, account for dynamic fitness landscapes and for the development of unforeseen or unexpected niches, can generate a climate for the generation of a complex system story formed by a society of agents based on interaction rich in complexity and creativity.

3 Embodiment

3.1 The Embodied Agent

'The affordances of the environment are what it offers the animal, what it provides or furnishes either for good or ill' [10]. Gibson postulates that our relationship with the environment is governed by our perception of it, which in turn is dependent on our bodily interactions within it. Different encounters in different environmental situations afford different qualities to different beings through interaction with their different bodies. Thus the way in which we perceive our environmental context is directly linked to our embodiment within it. Embodied perception of our environment is not just a case of situated experience, but rather an example of the deeper relation your body has on the understanding of the environment. For example, water for a pond skater affords a supportive surface where as for a human, due to its heavier body mass, it does not.

Interpretation of the environment can be considered to be directly linked to an agents embodiment within it, not just its level of embodiment physically, but what these sensed perceptions afford within its environmental context. Not only are agents situated in a dynamic environment, but also the way they perceive this environment is dependent on a combination of their embodiment within it and their bodily interpretation of it. The environment can thus be considered as a dynamic landscape within the system, possibly even as a dynamic agent in its own right. Environmental constraints themselves can be considered as an evolving fitness landscape with which the agents interact, thus allowing for a co-evolution of agent and environment [19], describing a complex relationship that never reaches a stasis.

3.2 The Embodied Viewer

A description of a viewers relationship with a generative artwork, can be a controversial one. There is one stance that suggests that the viewers appreciation of

the work comes about through an understanding of the beauty of the generative algorithm [5]. Other theories are that Evolutionary Art holds the attention of the viewer through comparability with models from nature [11], for others it is an emergence of apparently creative output from a computer that draws them in. Either way there is either a retinal [9]/sensory [13] or rational/cognitive [11] perception of beauty. Thus in both cases there is a strong relationship with how the viewer perceives the work (retinal), or how much of the process they perceive (cognitive) and what they get out of the work. Rather than considering the viewer as a voyeur, outside of the artistic process, listening in, removed from what is happening, looking down upon a virtual world as a kind of god like creature, omnipresent but impotent, exercising the notion of aura, described by Walter Benjamin as essential for the perception of art [3]. What if instead of this, the viewer is embodied within the same environmental constraints and space as the agents? In this scenario, due to a shared locality of stimuli, an understanding of the system space through sensual exploration becomes attainable. Rather than advocating a deeper level of immersion of the voyeur into the virtual world one could argue that this sharing of system-space is best achieved by introducing the agents into the same world as the perceiver i.e. the 'real' world. This facilitates a greater immersion in the piece with higher levels of communicability between viewer and agents, viewer and process and hopefully 'viewer' and music. The viewer can now be considered a participant in the system-space of the installation, another agent in the system. There is no need to model a virtual environment as the real world is an environment in its own right with more inherent richness that can be found in any simulation. This sharing of environmental context not only facilitates a direct communication between agent and participant but also encourages an ecology of communication to develop where an evolution of agent and environment includes the participant as an active member of society.

4 Enaction

The enactive [20][27] viewpoint builds on Gibson's ideas of 'direct perception' by doing away with the need for a Cartesian inner world model or representation. Rather than the more traditional perception - computation - action model there is now a more direct relationship between perception and action, with action informing the perception, which guides action, which informs the perception etc. [6]

This idea of a lack of an 'inner world view' or 'representation' such that all computation is a result of a direct relationship between action and perception has a commonality with the idea of agents and participants coexisting in real space. In both situations there is a move away from complex representations in simulated space to all interaction and perceptual representation happening in the real lived environment. This facilitates a shared mode of perception between agent and participant and affords the participant an enactive approach to the

perception of the generative process; all interaction is no longer modeled but happens through embodiment in a shared environment. The environment becomes the main catalyst for driving communication and creativity, each agent contributing to and taking from it like a shared cultural experience.

5 'Excuse Me': A Vehicle for Enactive Exploration

Installation is a medium that lends itself to an investigation of the viewer as an active participant within a shared system-space, another agent in the algorithm. There is already a concept within the medium of that the viewer is more than a voyeur, distanced from the work. 'The term 'looking' is superceded by the concept of 'spectating', which assumes a higher involvement by the audience [2] . There is also history in the field of installation of careful consideration of the relationship and boundaries, (or lack of), between viewer, work and its environmental context. Much of early Minimalist sculpture could be described as exploring ideas of subject/viewer, object/work relationships [4], relating to perception as set out by Merleau-Ponty in his publication 'The Phenomenology of Perception' [18]. Following Merleau-Pontys philosophy, subject and object are not considered to be separate entities; objects are considered to take their meaning from the way they are positioned in space and the way one views them through bodily interaction. In this context it is perhaps especially important to pay careful consideration to how the art works relates to the space it is situated in, the way it relates to or reinterprets this space and the way it is to be understood by the viewer. The installation 'Excuse Me' (2006), an interactive sound installation developed by the first author, attempts to set up a situation in which the notion of spectating in installation is superseded by this notion of participation in the piece, to such an extent that the spectator is not only interacting with the piece but becomes another agent in the artwork, an active participant in the creative process.

'Excuse me' tries to create the conditions for a deeper relationship between agent, participant and environment to develop. One in which there is a level of mutual exchange between the agents (including the participant) and environment; where environmental constraints can lead to unexpected outcomes; can account for dynamic fitness landscapes and for unforeseen niches to develop. For it is this relationship, formed on deeper embodiment of agents within their surroundings, with which we hope to generate a climate for richer results of complexity and creativity.

In 'Excuse me', the agents are placed in the same environment as the participants; they can communicate with this environment through listening and speaking devices, microphones and speakers. Allowing the participant and agents to coexist in the same environment and sonic ecology. Participants can have a direct affect on the ecology of the piece through sonic and physical interaction, either with an individual agent or with the environment as a whole.



Fig. 1. Photo of 'Excuse Me', Sonic Lab, SARC, June 2006

5.1 Installation Design

Each agent consists of a speaker and microphone element. The speaker element is constructed out of an audio transducer attached to the body of a violin; the microphone is a homemade electret with attached preamp. These microphones are placed somewhere on the body of the violin. There are six agents situated in the installation space, suspended from the ceiling, six being the minimum number considered by the author to be needed to form a society of sonic interaction. They are spaced apart with room for people to move amongst them and interact with them as individuals. The six agents listen to their environment and analyse the audio input to their microphone using Tristan Jehan's Max/MSP object, analyzer~ [12]. They then try to recreate their incoming aural stimuli through a matching process of this analysis of this audio to their own internal database of sounds. The microphone and speaker device of each agent is connected to a computer running Max/MSP [15]. After the agent has played back its 'best fit' sound to the environment it adds the analysis of the incoming sound to its internal database.

This re-sharing of an agents interpretation of the sound back into the environment allows for a culture of communication to develop, where the agents build up a library of shared sonic experiences. This kind of mimetic interaction is easy for the participant to appreciate and partake in. The algorithmic process is somewhat transparent and is only affected externally by environmental factors that are common to both agents and participant.

5.2 Subversion of Intention

A subversion of the mimetic intention of the agents is employed as a catalyst to the development of the sonic output. The system of analysis and recreation employed by the agents does not always give predictable results. For example the pitch-detecting algorithm is best suited to working in non-noisy, clean environments on monophonic sounds. Asking it to detect pitch of sound from a noisy microphone plugged into the environment, not only detecting one agents polyphonic output but an unpredictable complex output from up to five other

agents at once, plus possibly other extraneous environmental sounds, is pushing the software far beyond its intended limits. What do these machines make of the noisy surroundings they have been placed in, how will they interpret these situations where they are experiencing information overload? This added complexity is a benefit from the fact that the communication of these agents and their perceptual reconnaissance is all carried out in a real unpredictable environment, which leads to an evolution of un-predictable outcomes that could in turn be labeled creative. The environment could be labeled persistent as it is the individual agents intention to match the aural utterances of the others and their memorial internal databases that gives the system a notion of history. The environment acts as a medium of interaction between the agents allowing them to build a culture of communication, which is based on their own history of interaction and the interaction of the environment.



Fig. 2. Close up of Electret micropone element of one agent

Allowing the viewer to walk amongst these agents, listening to their individual output or listening to the society as a whole, whilst interacting on many different levels helps them to understand the internal (now external) algorithmic process. The agents pick up their environmental noise and try to recreate it; bits of speech and other sounds are broken up and reinterpreted, passed from one agent to another. Creating an evolving shared language of communication, a sonic ecology, an emergent texture of sound.

6 Conclusion

Through notions such as embodiment and enaction one can revisit design strategies for algorithms that pertain to operate in a creative environment. Weve argued that it is worth exploring strategies in which emergent and evolutionary algorithms inhabit an environment that is shared by a viewer/participant. We have questioned the role of the algorithm as model for delivering creative solutions directly and have proposed that both the algorithm and the viewer need to be situated and embodied in a shared environment. As exemplified in the installation 'Excuse Me', there is no attempt at creating a reductionist model of

the environment context but rather a placing of the algorithm and the viewer within the full complexity of the environment as a way of promoting unexpected complexities and interactions.

References

1. Almeida e Costa,F. & Rocha, L.M.: Embodied and Situated Cognition. Artificial Life. Vol 11 (1,2) MIT Press. pp 5 -11. (2005)
2. Benjamin,A (ed.): Installation art. Art & design profile ;no. 30. Art & design ;vol.8, no.5/6. London : Academy Group, (1993)
3. Benjamin, W.: The Work of Art in the Age of Mechanical Reproduction. 1935. Zeitschrift fr Sozialforschung (originally). www.marxists.org/reference/subject/philosophy/works/ge/benjamin.htm Accessed online (30 October 2006)
4. Bishop, C.: Installation Art, A Critical History. Tate Publishing, London. (2005)
5. Borevitz,B.: Super-Abstract: software art and a redefinition of abstraction. read_me Software Art and Cultures Edition, Goriunova,O & Shugin,A [eds] Center for Digital AEstetik-forskning, 2–4 pp 310-311. Also available online http://runme.org/project/+super-abstract/ accessed 29/10/06
6. Clark, A.:From Fish to Fantasy: Reflections on an Embodied Cognitive Science. A shortened and amended version appears as: "An Embodied Cognitive Science?" Trends In Cognitive Sciences 3:9:1999 pp 345-351. 1999 also online http://www.philosophy.ed.ac.uk/staff/clark/publications.html accessed 29/10/06
7. Dorin, A.: Beyond Morphogenesis: enhancing synthetic trees through death, decay and the Weasel Test, in Third Iteration: Proceedings of the Third International Conference on Generative Systems in the Electronic Arts, Innocent, Brown, McCormack, Mcilwain (eds) CEMA, Melbourne, Dec. (2005), pp119-128
8. Dorin, A.: The Sonic Artificial Ecosystem in Proceedings of the Australasian Computer Music Conference (ACMC), Haines (ed.). Adelaide, Australia. (2006). pp32-37
9. Duchamp,M. & Cabanne,P.: Dialogues With Marcel Duchamp. Da Capo Press. New York. (1987)
10. Gibson, J,J.: The Ecological approach to visual perception. LEA, London. (1986).
11. Jaschko. S.: Process as aesthetic paradigm: a nonlinear observation of generative art. generator.x conference, 23 - 24.September 2005, Oslo, Atelier Nord, (2005).
12. Jehan, T.: analyzer~ for Max/MSP available from http://web.media.mit.edu/ tristan/ accessed online on 29/10/06
13. Kant, I.: Observations on the Feeling of the Beautiful and Sublime. Trans. John T. Goldthwait. University of California Press, 1961, (2003).
14. Keil,D. & Goldin,D.: Indirect Interaction in Environments for Multi-Agent Systems. Wegnes,D., Parunak,V. & Michel,F. (Eds.). Environments for Multiagent Systems II. pp68 81. Spinger Press. (2006)
15. Max/Msp : http://www.cycling74.com accessed online on 29/10/06.
16. McCormack, J.: New evolutionary challenges for evolutionary music and art. Lanzi, P.I [ed] ACM SIGE V Olurion Newsletter Vol 1 (1) April 2006 pp 5- 1. (2006)
17. McCormack,J.: Open problems in Evolutionary music and art. Proceedings of Evo-Musart 2005, 30th March to 1st April Lausanne, Switzerland. (2005)
18. Merleau-Ponty,M.: The Phenomenology of Perception. London. (1998)
19. Moreno , A. & Etxeberria A.: Agency in natural and artificial systems Artificial Life Vol. 11 Issue 1-2 pp 161-176. (2005)

20. Noe, A.: Art as Enaction accessed online http://www.interdisciplines.org/artcog/papers/8/version/original (14/08/06)
21. Rozemond, M.: Descartes's Dualism. Cambridge: Harvard University Press (1998)
22. Smalley,D.: The Listening Imagination: Listening in the Electroacoustic Era. Contemporary Music Review 13:2. (1997).
23. Schafer,R.M.: The Soundscape: our sonic environment and the tuning of the world. Destiny Books, Vermont. (originally pub 1977). (1994).
24. Stevenson, I.: A Dialetic of Audible Space available online http://www.headwize.com/articles/steven_art.htm accessed on 29/10/06
25. Todd, P.M. & Werner, G.M.: Frankensteinian Methods for Evolutionary Music Composition. Griffith, N., and Todd, P.M. (Eds.) (1998). Musical networks: Parallel distributed perception and performance. Cambridge, MA: MIT Press/Bradford Books. 1998
26. Truax B. (2000).: Acoustic Communication, Greenwood Press, Westport, Connecticut 2nd edition. (2000) (1st Edition. Ablex Publishing, 1984)
27. Varela, F.J., Thompson, E. & Rosch, E.: The embodied mind : cognitive science and human experience. London ; Cambridge, Mass. MIT Press. . (1991)
28. Whitelaw,M.: System Stories and Model worlds: A Critical Approach to Generative art.. Readme 100 : Temporary Software Art Factory. (Norderstedt: BoD) pgs 135-154 Dec (2005).

Creating Soundscapes Using Evolutionary Spatial Control

José Fornari[1], Adolfo Maia Jr.[1,3], and Jônatas Manzolli[1,2]

[1] Interdisciplinary Nucleus for Sound Studies (NICS)
[2] Music Department
[3] Applied Mathematics Department
UNICAMP, Brazil
{fornari,adolfo,jonatas}@nics.unicamp.br
http://www.nics.unicamp.br/~fornari

Abstract. A new way to control sound spatial dispersion using the ESSynth Method is introduced here. The Interaural Time Difference (ITD) is used as genotype of an evolutionary control of sound spatialization. Sound intensity and the ITD azimuth angle are used to define spatial dispersion and spatial similarity. Experimental results where crossover and mutation rates were used to create spatial sonic trajectories are discussed.

1 Introduction

Biology has always inspired human creation, such as the relation of Soundscape to Sound Ecology. Here we correlate both to Evolutionary Computation (EC) [5]. Soundscape is the domain of acoustic design where a sonic environment is created by sounds [1]. As such, soundscape composition might aim to computationally emulate self-organized biological or natural acoustic environments [2]. In this work we propose the use of an evolutionary sound synthesis method, the ESSynth [3], for the dynamic creation and control of sound dispersion in soundscape. There have being several researches and applications of EC algorithms in the music domain. To mention a few: [6] uses genetic algorithms to study music evolution. [7] investigates the potential of artificial life (Alife) models for musical creativity. [8] uses artificial life algorithms to generate a multiplicity of possible artworks. [9] uses genetic algorithms for the design automation of new sound synthesis techniques. We have been also studying applications of Evolutionary Computation in Music [10, 11, 12, 13]. Recently, we have studied the usage of inter-aural time difference function (ITD) as a parameter for the sounds in the ESSynth population [14].

The term soundscape is attributed to Murray Schafer [1] and refers to an acoustic environment or an environment created only by sounds. There are three basic elements of the soundscape: *keynote-sounds*, *sound-signals*, and *sound-marks*. Keynote-sound is a musical term that identifies the key of a musical piece. The key might stray away from the original starting point, but it will eventually return. Comparing to a human population, keynote sounds may not always be consciously perceived, but they

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 517–526, 2007.
© Springer-Verlag Berlin Heidelberg 2007

outline the character of people living in a particular region, as they are created by nature, such as geographic localization and climate (wind, water, forests, birds, insects, animals). Even in urban areas there are keynote sounds, such as traffic sounds. Sound-signals are the foreground sounds, which are consciously listened to. (ex: warning devices, bells, whistles, horns, sirens, etc.). Sound-marks are sounds which are unique to a particular area. As Schafer pointed out, "*Once a Soundmark has been identified, it deserves to be protected, for soundmarks make the acoustic life of a community unique*".

Soundscapes are natural, self-organized processes usually resultants of an immense quantity of sound sources, correlated or not, but that conveys a unique audible experience that is at the same time recognizable and yet always original (as it never repeats itself). For this reason, creating a computation algorithm able to build an authentic soundscape is not trivial. For instance, stochastic processes, such as Markov chains, are not enough once that a random process can get astray of similarity. Deterministic processes are also not adequate because they fail to always deliver new sonic features. The human factor is therefore necessary to guide a process of soundscape design, steering it along the fine line between perceptual similarity and variation. Although soundscapes have several aspects that can be analyzed, here we are focused on the perceptually meaningful aspects of sound localization, once that a nontrivial sound localization scheme can lead to enticing soundscape effects. Here we present a method to control spatial sound dispersion using Evolutionary Computation. Our method explores the sound-marks dimension of a soundscape using two derived principles: *spatial dispersion* and *spatial similarity*. The idea is to use a inter-aural time difference function (ITD) to generate a trajectory of evolutionary sound-marks. To control this evolution we define a *spatial sound genotype* using two parameters: *sound intensity* and *ITD* azimuth angle, so we use genetic operators to reproduce new individuals in the population of sounds based on their spatial localization. Having the overall process running in real-time, it is possible to obtain as output an audio stream that is a continuous soundscape.

2 Evolutionary Sound Localization

As any other EC algorithm, ESSynth is primarily inspired by the Darwinian Theory of Evolution of Species [4,15], where biologic populations undergo evolution shaped by environmental conditions. As previously described in [11], ESSynth takes waveforms as individuals, represented by p(i,g), where each one is the i-th individual of the g-th generation's population, P(I,g)., that has a total of I individuals. The target set, T(J), also have individuals, represented by t(j), which are the ones that will guide the evolution. Figure 1 presents the evolutionary cycle of ESSynth. A more detailed description can be found in [17].

For the sound localization, initially we decided to work exclusively with the interaural time differences (ITDs) [18] because it has proven to be easy and less subjective to personal variations.

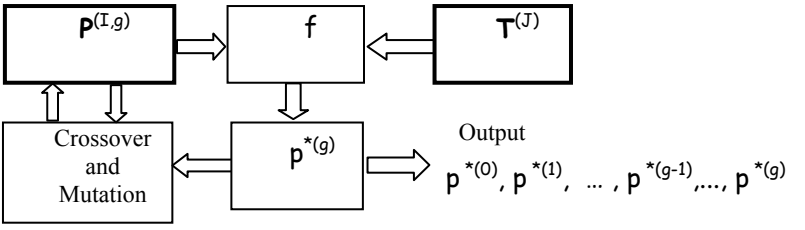


Fig. 1. ESSynth basic diagram

Where:

***P*$^{(I,g)}$*:** Population set with I individuals, in its g-th generation.*
***p*$^{(i,g)}$*:** the i-th Population's individual, in its g-th generation.*
***p*$^{(*,g)}$*:** best individual in the g-th population's generation.*
***T*$^{(J)}$*:** Target set of J individuals.*
***t*$^{(j)}$*:** the j-th Target's individual, responsible fort guiding the evolution.*
***f*:** fitness function.*

In order to describe spatial positioning using ITD, p(i,g) must now be described in two zero-padded channels, as seeing below:

$$p^{(i,g,d,\Omega)} = \begin{cases} \ldots, x_{n+d_1-1}, x_{n+d_1}, x_{n+d_1+1}, \cdots \\ \ldots, x_{n-d_2-1}, x_{n-d_2}, x_{n-d_2+1}, \cdots \end{cases} \quad (1)$$

Where Ω is the intensity, d_1 and d_2 are sample delays of each channel, given by zero-padding method. This implies in a phase-difference between channels that simulates the binaural cue given by the time difference that sonic waves take from its source to reach each ear of the listener. The ITD function is able to simulate only azimuth sound position, where elevation and distance are given by more sophisticated methods (although subject to personal distinctions), such as HRTF functions (Head-Related Transfer Functions) [19].

To calculate the azimuth angle of sound localization, we use the equation given in [18], relating the azimuth angle and the number of phase-differences.

$$\theta = \sin^{-1} \frac{(\Delta.\sigma).V}{c} \quad (2)$$

where: θ: azimuth angle with $\theta \in [0,180^\circ]$. $\Delta = \frac{1}{fs}$: time between samples. σ: number of delay samples. V: speed of sound, taken to be 384 m/s. c: distance between ears, taken as (the human average of) 0,25 m.

Taking $d_1=d_2$ in a symmetrical delay, the sample delay is now denoted by "d". As it is seen in the figure 2, the azimuth angle is related to the difference between the two paths from the source and from half of the distance.

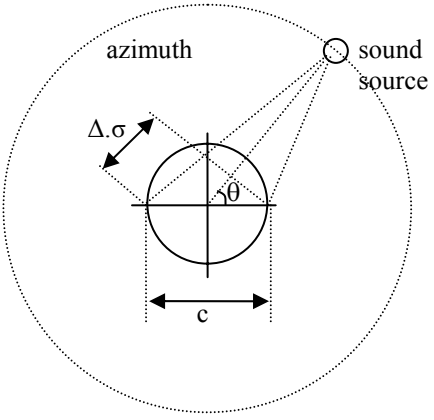


Fig. 2. Sound Spatial Positioning with ITD

From (1) and (2), the individuals, now embodying the sound localization parameter, are noted by $p^{(i,g,\theta_i,\Omega_i)}$, where, for each i-th individual:

$$\theta_i = \sin^{-1} \frac{3072.d_i}{fs} \tag{3}$$

3 Control of Sound Dispersion: Creating Sound-Marks

To generate and control the spatial evolution of individuals in the population, we created the *Spatial Dispersion Principle* and *Spatial Similarity Principle*; the associated two-dimensional parameters can be seen as the spatial genotype of the sound in the population. For simplicity, given a population $p^{(i,g,[2,N],\theta_i,d_i,\Omega_i)}$ described above we define a genotype of an i-th individual within the g-th generation of the population as S_i where i Є [1,N].

3.1 Spatial Dispersion Principle

1. For each individual a two-dimensional vector is associated, given by two scalars parameters: $S_i(A_i,L_i)$, where i=[1…N] is the number of individuals.
2. Parameter A_i Є [inf,+1] is the sound intensity, where Ω_{MAX} is the maximum intensity of all individuals, Ω_{MIN} is the minimum one, Ω_i is the intensity of the i-th individual, $Ai=\Omega i/\Omega_{MAX}$ and $inf= \Omega_{MIN}/ \Omega_{MAX,}$ which means that all intensities are normalized.
3. Parameter L_i Є [-1,+1] is the frontal-horizontal localization, given by the function $(90-\theta_i)/90$; where θ_i is the azimuth angle of the ith-individual.

3.2 Spatial Similarity Principle

Using the dispersion principle, the i-th individual with $S_i(A_i,L_i)$ is considered to be similar to another j-th one, with $S_j(A_j, L_j)$, if and only if, $L_i=L_j$ and $A_i=A_j$, for $i \neq j$.

Thus, the distance between the associated parameters $\Delta(Si,Sj)$ is given by:

$$\Delta(Si,Sj) = \frac{1}{2}\left|\frac{Ai-Aj}{1-\inf}\right| + \frac{|Li-Lj|}{4} \quad (4)$$

Note that the distance is normalized in [0,1] and it considers the proximity between intensity and the azimuth angle. Figure 3, presents the spatial principles used here in a drawing named as Sonic Localization.

Note that at the position (0,1), the sound will have the greatest intensity, and will be localized in front of the listener, which means, azimuth angle equals zero.

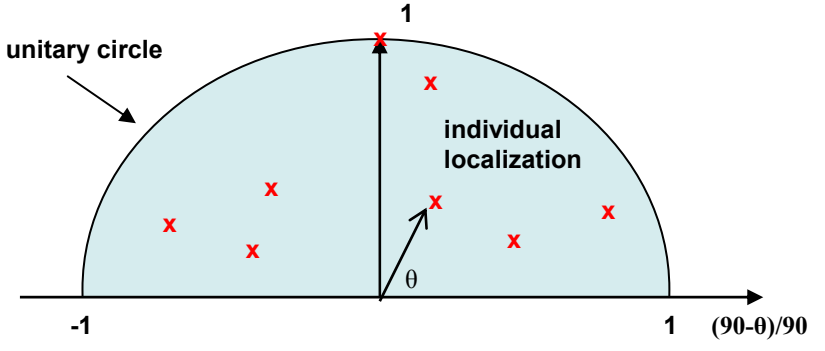


Fig. 3. The Sonic Localization Field

4 Genetic Operators

4.1 Crossover

Given $S^*(A^*, L^*)$ the best genotype of the g-th generation and the crossover rate ***α,*** where $0 \leq \alpha \leq 1$, all genotypes of the individuals in the population will be renewed by the following operation that is a convex combination:

$$A'_i = \boldsymbol{\alpha}.A^* + (1-\boldsymbol{\alpha}).A_i \quad \text{and} \quad L'_i = \boldsymbol{\alpha}.L^* + (1-\boldsymbol{\alpha}).L_i \quad \text{with } 1 < i < N \quad (5)$$

4.2 Mutation

Given $S_i(A_i,L_i)$, a genotype within the population, and the rate ß, where $0 \leq ß \leq 1$, the mutation operation produces the following transformation:

$$A'_i = ß.rand + (1- ß).A_i \text{ and } L'_i = ß\ .rand + (1- ß).L_i \text{ with } 1 < i < N \quad (6)$$

rand generates a random real value within [0,1] and the rate β controls the degree of randomness of the mutation operation.

Notice that the renewed values $S'_i(A'_i,L'_i)$ with $1 < i < N$ are the genotypes of the next generation (g+1)-th.

4.3 Target Set

The Target Set is constituted only by genotypes $T_j(A_j, L_j)$ and they can be organized around curve or as clusters of points within the Sonic Localization Field (see figure 3). This set is used to control sound dispersion location in the population therefore producing an evolutionary trajectory of sound-marks.

4.4 Fitness Function

To evaluate the similarity between the population and the target set, we define a Fitness Function $F:[0,1]\rightarrow[1,2,...,N]$ as:

$$F = \min_i \min_k \Delta(S_i, T_k) \tag{7}$$

with *i=1…N, k=1…M,* where Δ is the distance defined in (4) and T_k belongs to the genotype of the individuals in the Target set and S_i is the genotype of the i-th individual in the population of waveforms. Therefore, we denote the index of the best individual's genotype as $i^* \in [1,2,...,N]$.

During the evolutionary process only the sound with the best fitness, according to the spatial principles presented above, has its sound localization modified and projected into the sonic field. Given $S_{i*}(A_{i*},L_{i*})$ the best genotype of the g-th generation and according to the definition (3), the new individual of the next generation (g+1) is $p^{(i*,g+1,\theta_{i*},\Omega_{i*})}$ where $\theta_{i*} = 90.(1-L_{i*})$. $\Omega_{i*}=A_{i*}$. Ω_{MAX}.

5 Results and Discussion

We implemented a simulation of the model using Matlab. Our aim was to evaluate the potential of the method using the crossover and mutation operators as sound spatial dispersion controllers. Table 1 is presenting parameters used in three simulations discussed here.

Table 1. Parameters used in the simulations presented below, where G is the number of generations, N is the number of individuals in the population and M is the number of genotypes in the Target Set

Experiments	α	β
1	0.1	0.1
2	0.1	0.9
3	0.5	0.5
Fixed parameters: *G=50, N=100, M=10 and target set spatial dispersion = horizontal line.*		

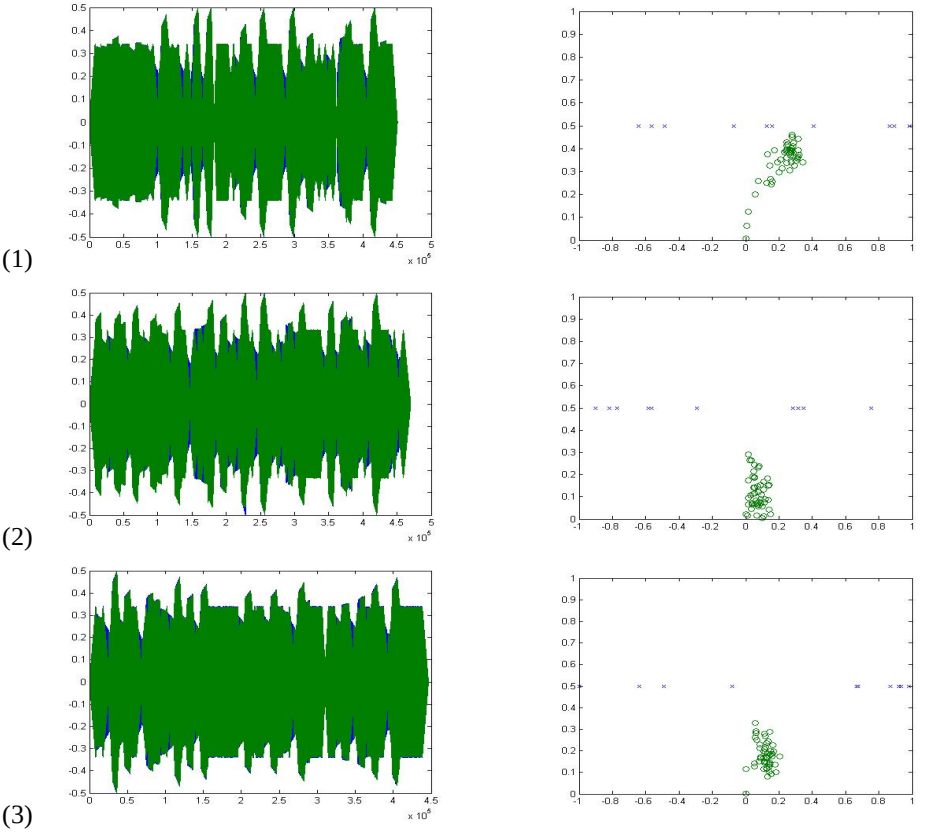


Fig. 4. Experimental results presenting the waveform and the genotype trajectory of the best individuals

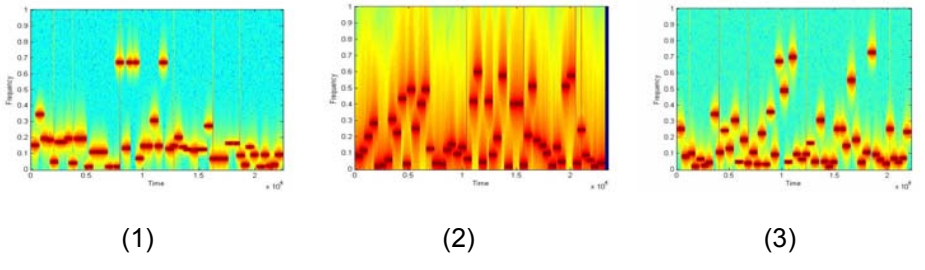


Fig. 5. Spectrogram of the three experiments shown in figure 4

The initial population of all experiments was a set of sine waves with frequency ranging from 80Hz to 4KHz, normalized with intensity=1 and azimuth angle = 0. So, this three waveforms above show how the method produced dynamic changes in the waveform amplitude envelopes (figure 4, right side), the genotype trajectory of the

best individuals (figure 4, left side) and spectral content produced by the overlap-and-add technique (figure 5). These aspects were coped to create an evolutionary trajectory of sound-marks.

It is possible to verify rich variations in the amplitude envelops that show how crossover and mutation rates were able to produce a wealth of dynamics in the time domain. All plots in the right side of the figure 4, describe a path in the sonic location field (figure 3), this behavior is related to the spatial dispersion of the sound and it is clear that the method is able to produce converging trajectories. Converging behavior associated with the amplitude diversity deliver to listeners the referred sound-marks.

Analyzing the spectrograms in figure 5, it is possible to understand the dynamic transformation produced also in the frequency domain. Since all initial sounds were sine waves within the same frequency range, it is clear that these three experiments generated unique spectral paths. They described a dynamic overlapping of sine waves that became a very interesting soundscape.

6 Conclusion

We presented here a new evolutionary method to control spatial dispersion of waveforms, producing evolutionary sound-marks in a soundscape. The results showed that it is possible to produce a rich amplitude envelop and interesting spectral behavior using dispersion and similarity principles through variations of crossover and mutation rates.

The synthesized sound was given by the queue of overlapped-and-added best individuals. Here, the resultant audio actually compounds a soundscape made by the perceptual convergence of best individuals as they get more similar along time, however, without being perceptually identical (clones). This asymptotic variant similarity delivers the kind of cognitive enhancement to synthesized sound that is hardly seeing in other synthesis methods. In this new approach, as the evolution goes, the best individuals are virtually allocated in different regions of space, by the ITD function. As the evolution process evolves in time, the generated soundscape keeps grabbing attention of the listener that is continuously trying to track the sound-mark trajectories.

As said in the introduction, soundscapes are formed by three basic elements: keynotes, sound-signals and sound-marks. Here we worked only with sound-marks as they are deeply related to sound localization. In further works we plan to research also evolutionary means of controlling keynotes and sound-signals. Both can be associated with the population's individuals content and its organization, respectively. Keynote-sounds could be given by a population of deterministic (quasi-periodic) individuals organized in a particular musical scale. Sound-signals could also be achieved by a population of stochastic (non-periodic) individuals behaving like background noises. Their genotypes would be built by other genotype descriptions such as partial distribution and pitch curves.

In future studies we also plan to implement a real-time application for this process, using Pd (PureData language) and test the usage of sensor-devices to capture the position of performers into the stage, such as cameras and infrared sensors, to produce geometric trajectories to build curves in the target set.

Acknowledgments

We thank the Brazilian sponsoring agencies that supported this research. Fornari is supported by FAPESP (www.fapesp.br) and Manzolli is supported by CNPq (www.cnpq.br).

References

1. R. Murray Schafer, M. (1977). “The Soundscape”. ISBN 0-89281-455-1.
2. Truax, B.. (1978) “Handbook for Acoustic Ecology”. ISBN 0-88985-011-9.
3. Manzolli, J., Fornari, J., Maia Jr., A., (2001) Damiani F. The Evolutionary Sound Synthesis Method. Short-paper. ACM multimedia, ISBN:1-58113-394-4. USA.
4. Miller, G. F. (2000). Evolution of human music through sexual selection. In N. L. Wallin, B. Merker, & S. Brown (Eds.), The origins of music, MIT Press, pp. 329-360.
5. Koza, John R., “Genetic Programming”, Encyclopedia of Computer Science and Technology. 1997.
6. Miranda, E. R.. “On the evolution of music in a society of self-taught digital creatures.” Digital Creativity, Vol. 14, No. 1, 29-42. 2003.
7. Soddu C. O. "A Generative Approach to Art and Design" Leonardo, 1 June 2002, vol. 35, no. 3, pp. 291-294. MIT Press. 2002.
8. Garcia, Ricardo A. "Automatic Generation of Sound Synthesis Techniques". Proposal for degree of Master of Science. MIT - Fall. 2000.
9. Gibbs, J. “Easy Inverse Kinematics using Genetic Programming”. Genetic Programming 1996: Proceedings of the First Annual Conference. MIT Press. 1996.
10. Moroni, A., Manzolli, J., Von Zuben, F., Gudwin, R., “Vox Populi: An Interactive Evolutionary System for Algorithmic Music Composition”, Leonardo Music Journal, San Francisco, USA, MIT Press, Vol. 10. 2000.
11. Fornari, J., Manzolli, J., Maia Jr., A., Damiani F., “The Evolutionary Sound Synthesis Method”. SCI conference. Orlando, USA. 2001.
12. Moroni, A. ; Manzolli, J. ; Von Zuben, F. J. . Artificial Abduction: A Cumulative Evolutionary Process. Semiotica, New York, USA, v. 153, n. 1/4, p. 343-362, 2005.
13. Von Zuben, F. J.; Caetano, M. F. ; Manzolli, J.. “Application of an Artificial Immune System in a Compositional Timbre Design Technique”. In: Jacob, C., Pilat, M.L., Bentley, P.J., Timmis, J.. (Org.). Artificial Immune Systems, Lecture Notes in Computer Science. Berlin: Springer-Verlag, v. 3627, p. 389-403. 2005.
14. Fornari, J.; Maia Jr. A.; Manzolli, J.. “A Síntese Evolutiva Guiada pela Espacialização Sonora”. XVI Congresso da Associação Nacional de Pesquisa e Pós-graduação (ANPPOM). Brasília. 2006.
15. Darwin, C. . The Origin of Species by Means of Natural Selection, or the Preservation of Favoured Races in the Struggle for Life.. ISBN: 1402171935. 1859.
16. Manzolli, J., Fornari, J., Maia Jr., A., Damiani F., “The Evolutionary Sound Synthesis Method”. Short-paper. ACM multimedia, ISBN:1-58113-394-4. USA. 2001.
17. Jack B. Kelly and Dennis P. Phillips. “Coding of interaural time differences of transients in auditory cortex of Rattus norvegicus: Implications for the evolution of mammalian sound localization”. Hearing Research, Vol 55(1), pages 39-44. 1991.

18. Murray, J. C., Erwin, H. R., and S. Wermter, "Robotic sound source localization using interaural time difference and cross-correlation", presented at KI-2004, September 2004.
19. Douglas S. Brungart and William M. Rabinowitz. "Auditory localization of nearby sources. Head-related transfer functions". The Journal of the Acoustical Society of America -- September 1999 -- Volume 106, Issue 3, pp. 1465-1479.
20. Fornari, J., Manzolli, J., Maia Jr., A., Damiani F.. “Waveform Synthesis Using Evolutionary Computa-tion”. Proceedings of the V Brazilian Symposium on Computer Music, Fotaleza. 2001.

Toward Greater Artistic Control for Interactive Evolution of Images and Animation

David A. Hart

Salt Lake City, UT, USA
http://dahart.com/

Abstract. We present several practical improvements to the interactive evolution of 2D images, some of which are also applicable to more general genetic programming problems. We introduce tree alignments to improve the animation of evolved images when using genetic cross dissolves. The goal of these improvements is to strengthen the interactive evolution toolset and give the artist greater control and expressive power.

1 Evolving Images

There have been many studies using different kinds of genotypes for evolutionary image synthesis and design. [2,3,4,6,7,10,11]. Perhaps the most well known image synthesis technique using interactive evolution was explored in Sims' seminal paper "Artificial Evolution for Computer Graphics" [10].

Sims used symbolic functions as genotypes, and simply evaluated the evolved functions at every pixel to express the genotypes into rendered image phenotypes. Symbolic functions are naturally represented as parse trees, with variables and constants at the leaf nodes, and parametric functions composing the internal branching structure. These tree structured genotypes start small and evolve complexity over time, as opposed to genetic algorithms that define fixed-length genotypes with a flat, linear structure.

1.1 Function Set

For design simplicity, we use a function set for genotypes consisting of only scalar valued functions. We have avoided the use of complex iterative operations, such as image processing operations, or pre-defined fractals.

Most functions in our set are purely functional, as opposed to procedural or imperative, but we do support imperative variable assignment for reasons we will discuss in section 2.5. Variable assignments are not allowed to participate in the selection process, however, because they introduce dependencies that are incompatible with cross-over mating and most mutation strategies.

For the images in this paper, we use arithmetic operators ($+$, $-$, $*$, $/$, $<$, $>$, $==$, $?:$), *sqrt*, *pow*, transcendentals (*sin*, *cos*, *asin*, *acos*, *tan*, *atan*, *exp*, *sqrt*, *log*), 1D-4D perlin noise functions (*noise*, *turbulence*), 1D curve functions (*lerp*, *smoothstep*, *linear*, *bspline*), and aliases a, r and *length* for polar coordinates. Several example genotypes and their corresponding phenotypes are pictured in figure 1.

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 527–536, 2007.
© Springer-Verlag Berlin Heidelberg 2007

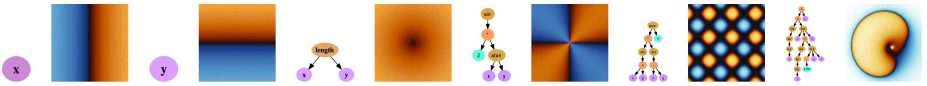


Fig. 1. Example genotypes and their expressed phenotypes: x; y; $length(x, y)$; $sin(sin(2 * atan(x, y)))$; $pow(sin(x + y) * cos(x - y), 3)$; $exp(sqr(log(r))) + lerp(r, (exp(sqr(log(r))) + lerp(-2.778, x, r)), y) + x$

1.2 Dynamic Range

Figure 2 illustrates a common issue with rendering symbolic functions: output values fall outside the range of displayable colors unless a compressing color mapping is used (consider $tan(x)$, for example).

We have found it useful to treat the output from symbolic functions as high dynamic range images, and apply any one of the standard tone operators to the image before display. The user can choose from a selection of tone operators.

In our experience one of the easiest to use has been Reinhard's "simple" operator $L/(1+L)$ [9], which smoothly compresses luminance values in the range $[0, +\infty)$ to $[0, 1]$. The user is of course given exposure and gamma controls.

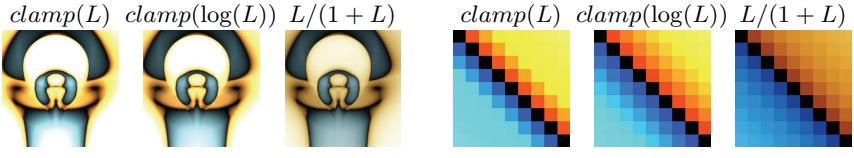


Fig. 2. Tone reproduction of symbolic functions

1.3 Mutations

Mutations on tree structured genotypes can be thought of as regular expression search and replace. For example, below we list the mutation types described by Sims [10] in loose regular expression form. Typically, all mutations are combined into an aggregate mutation by assigning each one a weighted probability, and normalizing the weights to one. In the example below, 'a' matches a node and its sub tree, 'n' matches constants, and 'v' matches variables. treeRand() constructs a random expression sub tree, and nodeRand() constructs a new node with given children.

$$
\begin{aligned}
Tree &: a \rightarrow treeRand() \\
Const &: n \rightarrow n + gaussRand() \\
Var &: v \rightarrow randVar() \\
Func &: a(a_1, a_2, \ldots) \rightarrow nodeRand(a_1[, a_2[, \ldots]*][, treeRand()[, \ldots]*]) \\
Node2arg &: a \rightarrow nodeRand([treeRand()[, \ldots]*], a[, treeRand()[, \ldots]*]) \\
Arg2node &: a(a_1, a_2, \ldots) \rightarrow a_{rand()} \\
Copy &: a \rightarrow treeCopy(getRandSubtree(getRoot(a)))
\end{aligned}
$$

1.4 New Mutation Types

Warp Mutations

$a \rightarrow a + treeRand();$ $\quad v \rightarrow v + const(gaussRand());$

$v \rightarrow v + treeRand();$ $\quad v \rightarrow nodeRand(v, treeRand());$

These are useful specializations of the $Node2arg$ mutation type. Examples are shown in figure 3.

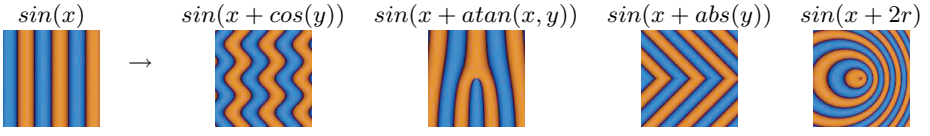


Fig. 3. Example warp mutations

Harmonic Mutations

$$T(x, y) \rightarrow T(x, y) + b * T(c * (x - j), d * (y - k))$$

Where sub tree $T(x, y)$ matches any function of x and y, any sub tree that contains x and y. b, c and d are random constants (typically $b < 1$ while $c > 1$ and $d > 1$). We often use $c := gaussRand() + 2$, $d := c$, and $b := 1/c$. j and k are optional random phase offsets. More generally, all variables in the sub tree can be mutated in the same fashion, making harmonic mutations applicable to all non-const nodes in the genotype.

This mutation represents the addition of a single harmonic. It is a way of introducing higher complexity and spatial frequencies into an evolving image while preserving the basic structure of the current image (see figure 4). Mandelbrot and IFS fractals are sometimes used to address this problem, though we prefer to avoid them only because their aesthetic styles can be so identifiable and visually dominant.

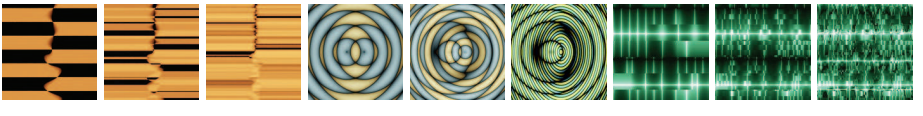


Fig. 4. Example harmonic mutations

1.5 Mutation Control

Providing user controls over mutation types, rates and biases has proven extremely useful, given that the human user's "fitness function" is constantly changing. Early in the process, one may want to make very large constructive changes ($Node2Arg$). Later when nearing a finished image, small mutations are desirable (Var, $Const$). In the middle of the process destructive mutations ($Arg2Node$) can become very useful for simplifying genotypes that are too complex and slow to render, or for distilling an image down to its basic characteristics. But it is not uncommon to want small changes early, or large changes late

in the process. Therefore, we feel control over mutations is best left to the user. Every automatic heuristic we tried inevitably reduced usability in some situations. Here are some simple user controls we have provided to address mutation control.

- Global relative mutation probability. A default of 10-25% seems to work well.
- Global absolute mutation rate, e.g. 2, 4, 8 nodes total.
- Control individual mutation types- allow the user to specify "var" mutations only for example.
- Control relative mutation rates for the aggregate mutation- user can change the probability for each mutation type, and the weights are re-normalized.
- Node height bias- mutation probability depends on a node's depth. High probability near the root leads to large changes, while high probability near leaves makes smaller adjustments.

1.6 Genotype Transforms

We provide the user with several genotype transforms that change the way genotypes respond to mutation. New variables or function definitions can be created from the current genotype. This expression aliasing is useful for strongly reinforcing the characteristics of an image in later images, without having to manually balance these characteristics throughout the selection process. When fine tuning is required at a later stage, the user can also expand these aliases back into primitives x and y.

Additionally a user can permeate a genotype with (initially benign) constants by inserting offsets and/or scaling factors in all but the const nodes: $a \rightarrow a + 0$; $a \rightarrow 1 * a$; $a \rightarrow 1 * a + 0$. Limiting mutations to constants is a good way to make fine-tuning adjustments. But often constants may not have evolved into every part of a genotype the user might wish to adjust. Inserting constants throughout the tree makes a genotype far more responsive to mutations on constants.

2 Alignments for Animating Evolved Images

Sims' genetic cross dissolve [10] is a technique that aims to blend or morph smoothly between two phenotypes by bridging similarities in the genotypes. Of all the animation techniques Sims describes, the genetic cross dissolve is the most analogous to traditional key framing of poses, and in practice is the easiest to use. Genetic cross dissolves have been used to animate linear fixed length genotypes, such as Draves' evolved fractal flames "Electric Sheep" [3]. This is fairly straightforward when using fixed length genotypes because there is a one-to-one mapping between the parameters.

Finding the similarities between tree structured genotypes, however, has traditionally been a difficult problem. Sims described the genetic cross dissolve as a technique to be applied to "two expressions of similar structure [...] by matching the expressions where they are identical and interpolating between the results where they are different. If the two expressions have different root nodes,

a conventional image dissolve will result. If only parts within their structures are different, interesting motions can occur." [10] Sims uses this same matching technique for cross-over mating.

2.1 First-Difference Alignment

The process of finding matches between two genotypes is called *alignment*. Alignment of genotypes is an entire sub field of computational biology [1,8]. We will refer to Sims' matching technique as the *first-difference alignment*. By aligning only the identical parts of the genotypes, and stopping at the first encountered difference, any and all similarities underneath are ignored. This can produce animation that is unrelated to the structural evolutionary similarities between the genotypes. In the worst case, when root nodes differ, the result is a static fade as Sims noted, rendering the dissolve ineffective as an animation tool. Unfortunately any two given genotypes are likely to have differing root nodes.

We believe this limitation is severe and unnecessary. There is no technical reason we cannot assign correspondences arbitrarily to non-matching nodes. In fact it is generally desirable to continue matching nodes after encountering a difference between the trees, since the result of the blend will often move rather than fade, as depicted in figure 5.

We now examine ways to relax the similarity constraint on the genetic cross dissolve. We first present several alternative algorithms for computing tree alignments. We then present a generalized evaluation procedure for genetic cross dissolves, using as input a tree alignment. Finally we show some visual results using these algorithms.

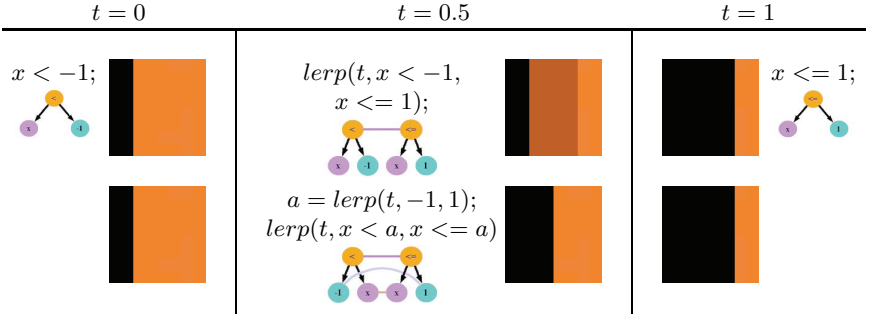


Fig. 5. A simple example of first-difference alignment (top) versus a full alignment (bottom), applied to two similar expressions with differing root nodes. The first-difference alignment produces a static region that fades over time, while the full alignment produces a moving boundary.

2.2 Leaf Node Alignment

A special case of the first-difference alignment is when both trees have identical node types and structure, and the only differences happen at the leaf nodes. In

this case, the alignment is a trivial one-to-one mapping, making it somewhat analogous to cross dissolves between fixed-length genotypes.

Leaf node alignments are particularly easy to use and are useful for animating, so to facilitate this type of alignment, we can construct one a-priori rather than search for it. A pre-aligned leaf node dissolve can be constructed by replacing all leaf nodes (constants and variables) in a tree with a blend node, initially blending each replaced leaf node with a copy of itself (e.g. $x \rightarrow lerp(t, x, x)$). Then to animate the leaf node alignment, mutation and selection proceeds with all operations restricted to modifying only the end point of any blend node in the tree (e.g. $lerp(t, x, x) \rightarrow lerp(t, x, y)$). This way, the user can first evolve a start frame, and then evolve the end frame directly from there, with the dissolve automatically built into the expression. With this approach, there is often no need to compute or examine animated sequences during the selection process, which can become painfully slow, only the end frame need be shown.

2.3 Constrained Alignment with Linear Cost

To explore full tree alignments, we first construct a simple ad hoc procedure to assign correspondences between the nodes of two trees. The nodes in a tree can be arranged into rows in a variety of ways. We use the distance from the root node as the node's row number. If both trees are arranged by row, we can assign a mapping between the rows, and then for each corresponding pair of rows, assign a mapping between the nodes, as depicted in figure 6. To guarantee that the dissolve produced using this alignment will be a valid tree, the root nodes are forced to match, and below the root we use an order preserving bijection for both the row mapping between trees and the node mapping between corresponding rows. The number of such bijections between two sets of size m and n, where n is size of the larger set, is $\binom{n}{m}$. We pick one of them at random.

In practice, this alignment performs well on trees that are very similar, but a single random alignment will rarely be optimal, especially for trees that are not very similar. Because of this, and because the alignment can be computed very quickly, we generate a large number of them, each time making different random choices for the row and node mappings. We then *compute a score for each random alignment*, based on the scoring scheme described in the next section, and present a small selection of the best alignments to the user for selection. This allows for some user control over the resulting animated dissolve.

The first advantage of the constrained alignment is in allowing user to select the alignment from a set of randomized choices, which has the very practical

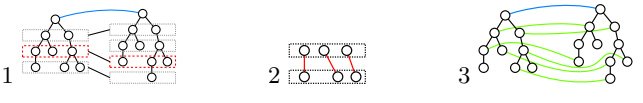


Fig. 6. Constrained alignment. 1) Align rows with a random, order preserving bijection. 2) For each matched row pair, align the nodes similarly. 3) The resulting alignment.

advantage of user control, which the other alignment techniques discussed here lack. Selection of the alignment also opens up the possibility of evolving new alignment criteria. The second advantage is its sheer simplicity- it is very easy to understand and code, and very fast to execute.

For displaying constrained alignments, we often evaluate only the midpoint (t=0.5) of the blend. Because the user has already seen both endpoints of the dissolve, seeing only the middle frame of the dissolve is usually enough to give a good sense of its quality.

2.4 Optimal Alignment with Quadratic Cost

Jiang et. al. [5] propose an algorithm for optimal alignment of ordered phylogenetic trees, and show that the cost for ordered and unordered binary trees as well as ordered n-ary trees of bounded degree is $O(|T_1| * |T_2|)$ where $|T_1|$ is the number of nodes in tree T_1.

We present a brief summary of their algorithm, and then outline some minor modifications needed in order to use this algorithm for our purposes. The basic idea is to examine all possibilities between a single pair of nodes that may have children, assign a score for each possibility, select the one with the best score, repeat recursively, and show by induction that the final score will be that of the optimal alignment. The alignment itself is computed using backtracking.

Consider figure 7. On the left we examine all possible ways to align a pair of nodes and on the right we examine every way to align a pair of forests of children, assuming the two parents are matched. Note that both of these operations are self-recursive and mutually recursive. It follows that every node in T_1 is compared against every node in T_2, and every forest of children of a node in T_1 is compared against every forest of children of a node in T_2. Intuitively this explains why the cost of the algorithm is $O(|T_1| * |T_2|)$.

The only change needed to align ordered binary trees is to remove the single out-of-order comparison from the forest matching case. The algorithm also easily extends to n-ary trees by expanding the two node-match cases in the forest match test to all possible injections (one-to-one mappings) of nodes between the two forests. Examining all possible injections includes allowing nodes on either side to align with nothing (called a "space"). Jiang et al's scoring metric rewards matching a node with another node, and penalizes matching nodes with spaces, to prefer alignments with fewer spaces whenever possible.

Jiang et al. describe a scoring measure μ as follows. Nodes have only a single attribute, called a label, designated "a", "b", etc.. Spaces are designated by "λ".

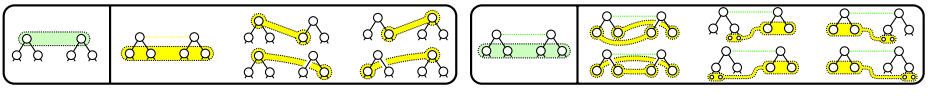


Fig. 7. Optimal binary tree alignment. On the left we consider all possible ways to match a pair of nodes. On the right we consider all possible ways to match the two forests of children of a pair of matched nodes.

They define $\mu(a,b) = 0$ when $a = b$, $\mu(a,\lambda) = 1$, $\mu(\lambda,a) = 1$, $\mu(a,b) = 2$ when $a \neq b$.

In our case, the label represents the node type (e.g. const, var, '+', sin, etc.,) but some nodes have an additional piece of data that must be considered separately. We want two different variables to be less likely to match than two of the same variable, and more likely to match than, say, a variable and a sine function. Therefore, we must modify the scoring measure. The $func$ node type refers to a user-defined function application, and its data is the function definition.

$\mu(a,b) = 0$ when $a = b$, and $a \notin \{const, var, func\}$
$\mu(a,b) = 0$ when $a = b$, and $a \in \{const, var, func\}$, and $a.data = b.data$
$\mu(a,\lambda) = 1$, $\mu(\lambda,a) = 1$
$\mu(a,b) = 1$ when $a = b$, and $a \in \{const, var\}$, and $a.data \neq b.data$
$\mu(a,b) = 2$ when $a \neq b$, or ($a = func$, and $a.data \neq b.data$)

We use unordered alignment for unordered binary nodes, and ordered alignment for ordered binary and n-ary nodes. Addition, for example, is an unordered operation, while subtraction is ordered. This mixed-order alignment scheme clearly still has $O(|T_1| * |T_2|)$ complexity.

The advantages of using optimal alignment are that we get the best correlation of similar structure in the genotypes, under criteria defined by $\mu(a,b)$. The disadvantages include high memory and time complexity, and the lack of user control or the option for selection of the alignment. Examples of this alignment applied to evolved genotypes are shown in figure 10.

2.5 Using an Alignment to Contruct the Dissolve Expression

After we align two expression trees, we then need to be able to evaluate it as a dissolve. Sims described a specialized evaluation procedure for the first-difference alignment which traverses the identical parts of both trees simultaneously, but stops descent and performs an interpolation between any differing nodes. [10] Full alignments cannot use this evaluation procedure since they typically cannot be traversed simultaneously.

We instead use the alignment to connect both trees into a single explicit expression for straightforward evaluation. This can be done by creating a blend

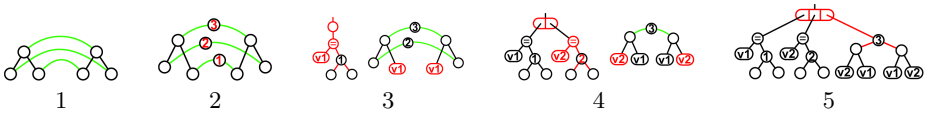


Fig. 8. Using an alignment to construct the imperative evaluation cross dissolve. 1) Align trees and create a blend node for each match. 2) Sort the matches from bottom to top. 3) Cut the bottom-most blend sub-tree and replace connections to it with a new variable. Create a new variable assignment expression and connect the blend sub-tree that was just cut. Append the new variable assignment expression to the end of an ordered list. 4) Repeat step 3 until there is only one blend node left. 5) Append the final blend node to the end of the ordered list. The list's "value" must be defined as the value of the last item in the list.

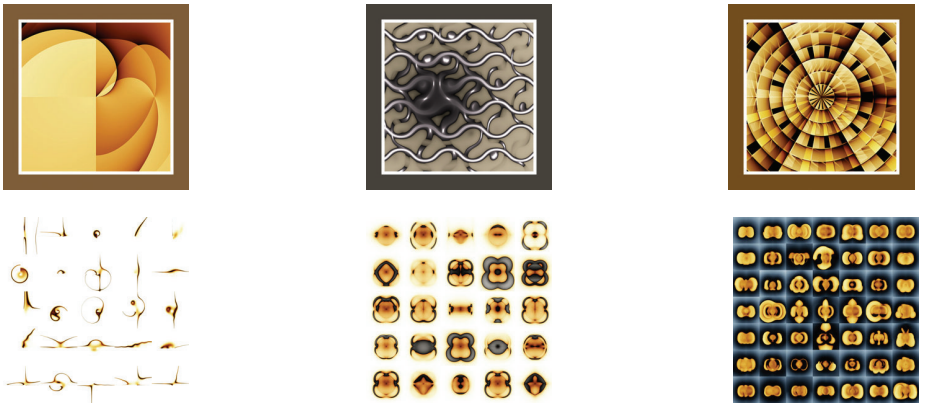


Fig. 9. Top, fully evolved images. Bottom, populations of mutated variations.

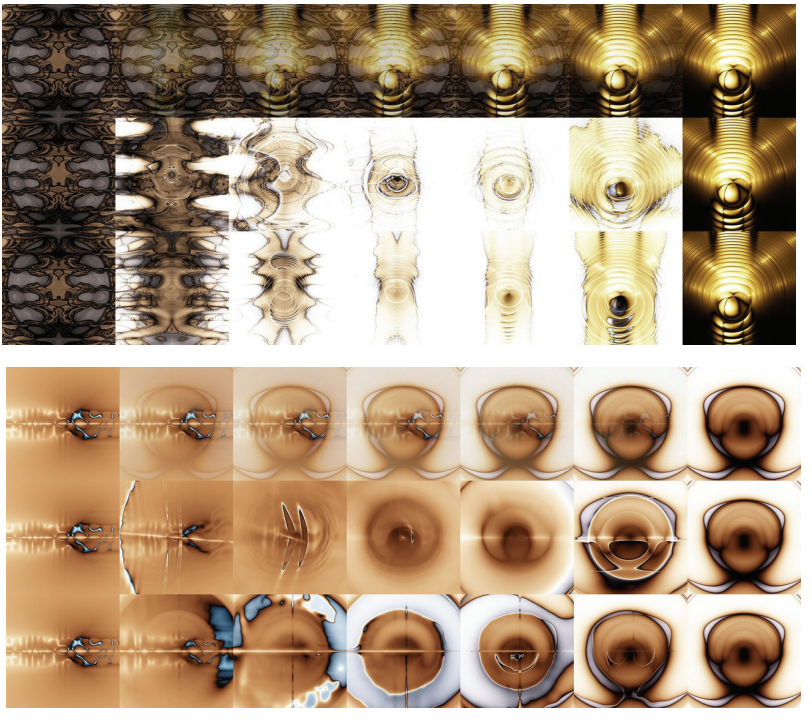


Fig. 10. Two examples of genetic cross dissolve sequences produced using different alignments on two genotypes which did not explicitly share heritage. Top row: First-difference alignment. Only the root nodes were connected, resulting in a static image fade between genotypes. Middle row: Constrained random alignment. Similar to optimal alignment, but with a little bit of (perhaps desirable) random variation. Bottom row: Optimal alignment. This gives us the most visual insight into the structures that are shared between the two genotypes.

node for each pair of matched nodes in the alignment, creating a blend node between the root nodes, and then rewiring the tree to allow for imperative evaluation, as depicted in figure 8.

After the dissolve has been generated, we apply a final optimization step to speed evaluation and reduce redundant nodes by traversing the tree and collapsing blend nodes and their two children into a single copy of one of the children, in the case that both child nodes and their sub-trees are identical.

3 Conclusions and Results

We have presented several tools that improve the artistic controllability of interactive evolution. Two new expressive mutation types were given, and genotype transforms were introduced as a way to make genotypes more receptive to mutation.

We have eliminated previous limitations when blending between evolved images by introducing tree alignment into the genetic cross dissolve process. We have created an algorithm for constructing the dissolve expression using an alignment, and presented several alignment methods, each with different advantages.

References

1. P.J. Bentley and J.P. Wakefield. Hierarchical Crossover in Genetic Algorithms. *Proceedings of the 1st On-line Workshop on Soft Computing (WSC1)*, 1996.
2. R. Dawkins. The Blind Watchmaker. *Harlow Logman*, 1986.
3. S. Draves. The electric sheep screen-saver: A case study in aesthetic evolution. *Applications of Evolutionary Computing, LNCS*, 3449, 2005.
4. G. Greenfield. New Directions for Evolving Expressions. *Bridges: Mathematical Connections in Art, Music, and Science*, pages 29–36, 1998.
5. T. Jiang, L. Wang, and K. Zhang. Alignment of trees- an alternative to tree edit. *Theoretical Computer Science*, 143(1):137–148, 1995.
6. C.G. Johnson and J.J.R. Cardalda. Genetic Algorithms in Visual Art and Music. *Leonardo*, 35(2):175–184, 2002.
7. M. Lewis. *Creating Continuous Design Spaces for Interactive Genetic Algorithms with Layered, Correlated, Pattern Functions.* PhD thesis, PhD thesis, Ohio State University, 2001, 2001.
8. C. Notredame. Recent progresses in multiple sequence alignment: a survey. *Pharmacogenomics*, 3(1):131–144, 2002.
9. E. Reinhard, M. Stark, P. Shirley, and J. Ferwerda. Photographic Tone Reproduction for Images. *Proceedings of SIGGRAPH 2002*, pages 267–276, 2002.
10. K. Sims. Artificial evolution for computer graphics. *Proceedings of the 18th annual conference on Computer graphics and interactive techniques*, pages 319–328, 1991.
11. S. Todd and W. Latham. The mutation and growth of art by computers. *Evolutionary Design by Computers*, pages 221–250, 1999.

Evolutionary Assistance in Alliteration and Allelic Drivel*

Raquel Hervás[1], Jason Robinson[2], and Pablo Gervás[1]

[1] Dep. Ingeniería del Software e Inteligencia Artificial,
Universidad Complutense de Madrid, Spain
raquelhb@fdi.ucm.es, pgervas@sip.ucm.es
[2] University of Georgia, USA
srjrobin@uga.edu

Abstract. This paper presents an approximation towards an evolutionary generator of alliterative text. A simple text is given along with the preferred phoneme for alliterations as input. Then the evolutionary algorithm (with the aid of a phonemic transcriber, Microsoft Word and Google) will try to produce an alternative sentence while trying to preserve the initial meaning and coherence. A bigram language model and the evaluation of the phonetic analysis are used to assess the fitness of the sentences.

1 Introduction

Alliteration is a literary figure difficult to explain in precise terms. For example, some human poets consider alliteration to be the repetition of consonant sounds at the beginning of words; others consider it to be the repetition of consonant sounds in stressed syllables; and others may consider it to be the repetition of any sounds anywhere in the utterance. In any case, alliteration involves repetition of sound within a text of certain phonetic ingredients. However, feasible as it may be to carry out the phonetic analysis of an utterance, there is an intrinsically aesthetic component to the perception of alliteration by humans which is difficult to reduce to rules. A large number of ingredients seem to play a role in the process - certainly a repeated occurrence of certain phonemes, but also avoidance of radical departures from an intended meaning, rhythm, whether the given phoneme starts a word, similarities between the sound of the repeated phonemes and the sounds associated in real life with the concepts being mentioned... Moreover, the magic of a good alliteration lies in finding a perfect balance between this set of ingredients. We believe that an evolutionary solution - with possible modifications to a text modelled as mutation operators, and the ingredients that must be judged represented as fitness functions - is a good method for exploring the potential of applications for automated alliteration of input sentences. This

* Partially supported by the Spanish Ministry of Education and Science project TIN2006-14433-C02-01.

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 537–546, 2007.
© Springer-Verlag Berlin Heidelberg 2007

paper presents an initial exploration of this problem, using synonym substitution as an elementary modification mechanism, *synset* preservation as means of avoiding drastic shifts in meaning, and a combination of an n-gram language model and evaluation of phonetic analysis to evaluate the fitness of individuals. The n-gram language model is based on Google searches, calculating the probability of occurrence of a bigram in terms of its frequency of appearance as a Google search.

The current project isolates alliteration, in its broadest definition, in order to provide an inchoate step in a process that could eventually incorporate many other stylistic weapons in a writer's arsenal (be the writer human or machine). Once we have proven algorithms that generate semantically similar sentences with greater occurrences of alliteration, we can then apply this algorithm to other stylistic possibilities such as rhythm and rhyme and even more difficult semantic tricks.

2 A Strange Brew

This section outlines the combined use of typical research methodologies - Linguistics and Evolutionary Algorithms - with widely available commercial tools like Microsoft Word and Google.

2.1 Phonetics and Alliteration

The importance of alliteration in written Spanish style has often been considered inconsequential, deprecated or more a facet of germanic languages than it is of Spanish. Nevertheless, if one were to subscribe to theories such as Roman Jakobson's, which states that corresponding sounds suggest correspondence in meaning [1], alliteration would be a valid tool for all authors, regardless of language.

A valid phonemic (or phonetic) transcription is necessary for any algorithm that will be capable of measuring phonic qualities of an utterance in most languages. In Spanish it is true that a phonemic transcription is not as necessary for identifying alliteration as it would be for English due to the high correlation between phonemes and graphemes in Spanish; however, to evaluate rhyme and rhythm, the role of the phonemic transcription will become more critical.

An automatic phonemic transcription in Spanish is straightforward by following standard orthographic [2] and phonemic conventions [3,4]. The universal transcription algorithm used in this work was originally developed for a historical research project on Spanish syllabification, and revised and augmented in various projects since [5]. It will have some limitations namely dialectal differences and loan words. Nevertheless, the authors of this work have found that automatic transcription produced by this algorithm is sufficient for the majority of Spanish words, and therefore valid for this study.

Our computer generated transcription translates letters into phonemes and then groups these phonemes into syllables. Once this has been done, the theoretical stress can be calculated according to well defined Spanish grammatical

rules [6,3]. Note that the new transcription output will look very similar to the original Spanish spelling, because Spanish is so much more phonetic than other languages such as English or French.

By default the phonemic transcription used in this project includes many other features such as syllabification, accentuation and various statistics functions. The statistics functions return counts of syllables, words, unique words, individual phonemes and total phonemes. Many of these counts will be utilized by the evolutionary algorithm.

2.2 Synonyms in Microsoft Office

Who loves to hate Microsoft? A powerful and primary programmatic element of our project is a COM[1] interface with Microsoft Word. Microsoft has invested considerable time and money into the linguistic capabilities of their products. No matter what stance the readers may have towards this company and its products, we wanted to illustrate the tremendous benefit that one can realize by reusing what is already available as opposed to reinventing the wheel. Furthermore, the ability to lookup synonyms and antonyms for almost any given word in almost any given language seemed beneficial enough to justify the use of this common commercial product.

Our function that accesses this benefit is less than 100 lines of C++ code and is flexible enough for us to change the language for which synonyms are retrieved with one single parameter. An example in Spanish shows how Microsoft's API returns a series of synonyms for each word queried. These series, though less structured and consistent, are somewhat analogous to Wordnet's *synset* [7]. In this case a search for "pago" will return (as formatted by our interface):

- [wdNoun, desembolso (desembolso, reembolso, cancelación, liquidación, dispendio, entrega)]
- [wdVerb, pagar(pagar)]
- [...]

These sets of synonyms are organized by their grammatical function, which can be filtered. The importance of this is obvious: if you are replacing a noun in a sentence with a synonym, you do not want to replace it with a verb.

2.3 Statistical Language Modelling n-Gram Models

Statistical Language Modelling (SLM) [8] is the attempt to capture regularities of natural language for the purpose of improving the performance of various natural language applications. By and large, statistical language modelling amounts to estimating the probability distribution of various linguistic units, such as words, sentences, and whole documents.

Ironically, the most successful SLM techniques use very little knowledge of what language really is. The most popular language models (n-grams, that are

[1] Component Object Model.

consecutive sequences of n words in a text or sentence) take no advantage of the fact that what is being modelled is language - it may as well be a sequence of arbitrary symbols, with no deep structure, intention or thought behind them.

2.4 Evolutionary Algorithms

We propose the use of evolutionary algorithms (EAs) to deal with the generation of alliterations in Spanish. In the case under consideration, the main advantage we can find in evolutionary algorithms is that they do not need specific rules to build a solution, but rather a form of quantitative evaluation.

Evolutionary techniques have been shown in the past to be particularly well suited for the generation of verse. The work of Manurung [9] and Levy [10] proposed different computational models of the composition of verse based on evolutionary approaches. In both cases the main difficulty lay in the choice of a fitness function to guide the process. Although Levy only addressed a simple model concerned with syllabic information, his overall description of the architecture in terms of a population of poem drafts that evolve, with priority given to those drafts that are evaluated more highly, is an important insight. The work of Manurung addresses the complete task, and it presents a set of evaluators that grade the candidates solutions according to particular heuristics. An important conclusion to draw from these efforts is the suitability of evolutionary techniques for natural language generation tasks in which the form plays a significant role, to the extent of even interfering with the intended content.

3 An Evolutionary System for Alliterations

The work presented here is intended to be an evolutionary generator of alliterations from a given sentence and the phoneme that must be repeated in the intended alliteration. The operation of the system is based on the phonetics of the initial sentence and the use of synonyms for the words in the phrase. The algorithm produces a solution phrase with a similar meaning than the initial one, but formed by words containing the specified phoneme.

In this evolutionary algorithm, each word of a sentence is treated as a gene. The initial population is generated randomly, using for each word from the initial text a synonym obtained from Microsoft Word. The system works over this population for the number of generations determined by the user. In each generation two typical genetic operators are used: crossover and mutation. Finally, at the end of each generation each sentence is evaluated and a selection of the population is passed to the next one, in such way that the phrases with a higher fitness value have more possibilities of being chosen.

3.1 Data Representation and Genes

The population with which the system operates is made up of various instances of the sentence that has to be alliterated. The words of this phrase are considered to

be the genes. Each word of the sentence has an associated tag indicating its part of speech. So, the synonyms searched for are only the ones that correspond to the same part of speech as the corresponding word, thus avoiding inconsistencies.

When looking for the synonyms of a given word, Microsoft Word returns a list of synonym sets or *synsets*. To avoid significant departures from meaning, a unique *synset* is considered for each word of the initial sentence. The mutations of each word will be carried out with this stored *synset*, including the initial word if it was not in the first *synset*.

An example of input sentence could be

```
"El sonido con que rueda la profunda tormenta"
```

It is an alternative version of the well-known alliteration from José Zorrilla *"El ruido con que rueda la grave tempestad"*. The associated part of speech tags of this sentence are

```
[other,noun,preposition,conjunction,verb,other,adjective,noun]
```

3.2 The Genetic Operators

Two genetic operators have been used: crossover and mutation.

For the crossover operator, two sentences are selected randomly and crossed by a random point of their structure, so that each of the children will have part of each of the parents' traits.

In the case of the mutation operator, some of the genes are chosen randomly to be mutated. If a word is selected to mutate, the system asks Microsoft Word for synonyms corresponding to the same part of speech as the word to mutate. One synonym is randomly selected to substitute the previous word. The *synsets* are used as described in Section 3.1.

A special case in the mutation are the articles "el", "la", and their corresponding plural forms. Even though they do not have synonyms, it is necessary to exchange them with their opposite gender form. This is because the *synsets* for nouns do not distinguish between the feminine and masculine gender. In order to obtain coherent results in the final sentences we have to mutate the articles from "el" to "la" and vice versa, using the same mutation probability applied to the rest of words.

3.3 The Fitness Function

The key to the evolutionary algorithm lies in the choice of the fitness function. A sentence in the population is considered better or worse depending on two factors: the number of appearances of the desired phoneme and the coherence of the sentence.

The phonemic fitness value of a phrase is calculated as the percentage of appearances of the phoneme out of the total number of phonemes in the phrase[2]. In

[2] Even when the common definition of alliteration requires the target phoneme to appear at the start of the word, there are some exceptions. We have chosen not to take into account the position of the phoneme in this approximation.

the extreme, the maximum value for a sentence would be 100% if every phoneme in the sentence were the same. Of course this would be impossible, but that measure is enough to study the resulting phrases provided by the evolutionary algorithm.

The coherence fitness value is calculated using bigrams. For each pair of consecutive words in the sentence we query Google to get the number of appearances of the pair in the Internet. This value is normalized using the number of appearance of each of the words separately compared to the number of times they appear side by side, as shown in Formula 1, and the coherence of all the bigrams is sumed up to obtain the coherence fitness of the whole sentence.

$$coherence(w1, w2) = app(w1 + w2)/(app(w1) + app(w2)) \quad (1)$$

The final fitness value for the whole sentence is the sum of the phonetic and the coherence fitnesses. With this measure, we are trying to obtain the most alliterations in a phrase with the minimum loss of coherence.

3.4 Example of Execution

For the operation of the algorithm three linguistic inputs and four genetic parameters are required.

For linguistic inputs we consider the initial sentence that is going to be alliterated, the tags indicating the part of speech of each word in the sentence, and the phoneme with which we desire to alliterate. The genetic parameters required are the number of individuals in the population, the number of generations to be executed, and the crossover and mutation operator probabilities.

Considering the example and the initialization exposed in Section 3.1, some sentences from the initial population could be

```
El sonido con que vira la recóndita precipitación
El sonido con que gira la oscura inclemencia
```

For the crossover operator pairs of sentences are selected randomly to be crossed by a random point of their structure. In this example, if the previous sentences are crossed by the fifth word, the result could be the following:

```
El sonido con que vira la oscura inclemencia
El sonido con que gira la recóndita precipitación
```

For the mutation operator, some words from the sentences are randomly chosen to be mutated. Mutation consists then in substituting the word for one of its synonyms from Microsoft Word, always corresponding to the same part of speech. In this example, if the word "sonido" from the first sentence is selected to mutate, we must choose its synonym from the stored *synset* provided by Microsoft Word:

```
[eco,resonancia,retumbo,cacofonía,asonancia,eufonía,fama,monotonía]
```

In this example, the chosen word has mutated to "retumbo", and it is the only word that mutates.

After crossover and mutation the fitness values for the two sentences of the example are given in Table 1. This table also includes the fitness values of the original alliterative text by José Zorrilla in order to compare our output with one "ideal" result. We are supposing that the searched phoneme is *'r'*. For the coherence fitness, the sentences are evaluated using bigrams. If we take as example the pair of words "El retumbo", Google returns a count of 35.100.000 for the word "El", 80.000 times for "retumbo" and 1.390 times for both words as a phrase. So, for this pair of words the coherence fitness is 3,95E-5.

Table 1. Fitness values for examples of sentences

Sentence	Appearances of phoneme	Total number of phonemes	Phoneme Fitness	Coherence Fitness	Total Fitness
El retumbo con que vira la oscura inclemencia	2	37	5.41%	0.89%	6.30%
El sonido con que gira la recóndita precipitación	3	41	7.32%	4.92%	12.24%
El ruido con que rueda la ronca tempestad	2	33	6.06%	4.37%	10.43%

4 Experiments and Results

To test the feasibility of the idea of alliterating words within a sentence using EAs, we have carried out experiments taking the initial sentence "El sonido con que rueda la profunda tormenta" and setting this as our goal. We have executed several experiments using different population sizes and number of generations. The crossover and mutation probabilities are constant in all the experiments. The searched phoneme is "'rr".

In Table 2 we can see the numerical results of the experiments, and in Figure 1 its graphical representation. This chart illustrates the correlations between the rise in population size and the rise in phonemic fitness with the decrease in coherence fitness regardless of the number of iterations per example sentence. For each resulting sentence the graph shows the number of individuals and iterations, indicated by the bar on the left, to be measured against the fitness measures on the bar to the right. The phonemic fitness, the line plotted with squares, rises as the population size rises; while at the same time the coherence fitness drops, denoted by the triangle plotted line.

With small populations (10 and 25 individuals) the coherence and phonemic fitness values are quite similar. Sometimes the coherence fitness value is greater; sometimes the phonemic. However, as the population size increases the phonemic fitness raises while the coherence drops. This means that the individuals with the most total fitness are the ones with higher phonemic fitness, and are not

Table 2. Table of numerical results

Population size	Number of generations								
	10			25			50		
	Phon. Fitness	Coh. Fitness	Total fitness	Phon. fitness	Coh. fitness	Total fitness	Phon. fitness	Coh. fitness	Total fitness
10	3,03%	5,17%	8,20%	5,56%	4,60%	10,16%	5,56%	5,19%	10,75%
25	8,11%	4,63%	12,74%	8,11%	4,63%	12,74%	8,11%	4,63%	12,74%
50	10,53%	0,40%	10,93%	10,81%	0,40%	11,21%	10,47%	0,35%	10,81%
100	10,37%	0,44%	10,80%	10,46%	0,39%	10,85%	10,53%	0,41%	10,93%
300	10,53%	0,41%	10,93%	10,81%	0,40%	11,21%	10,81%	0,40%	11,21%

necessarily good from the point of view of coherence. This problem can be seen easily in Figure 1, where both fitness values start more or less at the same level, but as the phonemic increases the coherence tends to decrease. In our first approximation both coherence and phonemic fitness values have equal weights in the total fitness, but the algorithm should be adjusted to find the proper weights for each one and improve the results.

From the point of view of the coherence fitness, the values obtained are quite low. The maximum value for a pair of words in the sentence would be 1 if the number of appearances of both words together is equal to the sum of the appearances of each of the words separately. This is almost impossible. In addition, when one of the words is a very common one - for example "el" -, the number of appearances is much higher than the number of appearances of the bigram, provoking low values for the coherence fitness.

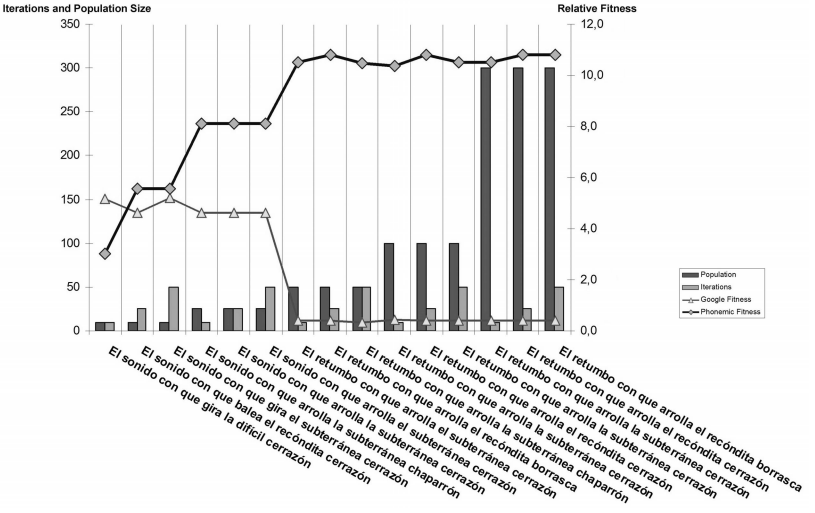


Fig. 1. Graphical representation of the results

In addition, we have also obtained strange results from the point of view of coherence. For example, in a generation of the system, we found that the sentence "El sonido con que gira el subterránea cerrazón" has more coherence fitness than "El sonido con que gira la subterránea cerrazón", which is the most correct one. This problem is due to the fact that we are normalizing the fitness of each pair of words and then summing. This means that pairs of words that have many appearances according to Google are considered to have the same weight as the combinations with fewer appearances. In this case, the fact that "gira el" is more common than "gira la" is enough to give more fitness to the first sentence even when "la subterránea" has many more appearances than "el subterránea". It is clear that we must study ways to prevent this type of incongruence.

5 Conclusions and Future Work

Our experiment with evolutionary algorithms and alliteration denotes obvious strengths and weaknesses.

An important difficulty in the task of generating alliterative sentences is the fact that it requires taking into account information at different levels of linguistic competence, such as phonetics, morphology, syntax and semantics. In accepted examples of good alliteration, the information at these different levels interacts in non-trivial ways to produce the desired literary effect. Although the evolutionary approach presented in this paper is still far from reproducing the competence of even a mediocre human writer, it has shown that a combination of mutation operators and fitness functions of varying complexity might be a manageable way of modeling these interactions computationally. This we consider the most important strength of the approach. Given the complexity of the interactions identified in examples from the literature, it seems improbable that acceptable results be obtained by means of simpler algorithmic or rule-based approaches. Of our experiment we are most proud of the broad, flexible base that we have laid and the promise that this base holds for interesting future work. The weaknesses, though not discouraging, have illustrated some areas that we need to improve. Many of these are in the process of being addressed.

The addition of the Google searches did improve the overall semantics slightly. Further refinement of the Google searches - for instance by extending it from bigrams to trigrams - should improve the quality of the sentences. The search "el difícil" and "difícil fama" may prove frequent, incorrectly validating sentences like "el difícil fama". This would be avoided using trigrams. On-line queries of another more precise Spanish corpus such as [11] might also yield better results.

The addition of a morphosyntactic disambiguator may help providing the part of speech tags as input of the system and checking output sentences for overall validity. We may implement our own, or simply query The Grupo de Estructuras de Datos' Morphosyntactic Disambiguator online in [12].

For the computational evaluation of sentence quality we considered two variables, alliteration versus frequency, with equal weight. This may be refined. Most people prefer a completely mundane sentence that makes sense to line of

alliterative drivel. Our future work will strive to find an acceptable balance between these extremes, and then apply this balance to the evolutionary algorithm.

In this work we have considered that an alliteration is based in the repetition of a phoneme, but it can also be provoked by the repetition of syllables. Also we can consider apart vowels and consonant, or even the repetition of two or more phonemes.

More mutation operators can also be added. For adjectives, to add another adjective with the same meaning is a good option. For instance, "la grave ronca tempestad" is a very common structure in Spanish. In the same line, another operator that duplicates structures can also provides interesting results. For instance, "rueda la profunda tormenta, grave tempestad", where the pair adjective-noun has been duplicated.

Once the aforementioned weaknesses have been attended to, we will then apply what we have learned to more patterns in language other than alliteration, such as rhythm and rhyme.

References

1. Jakobson, R.: Language in Literature. Cambridge: Belknap Press of Harvard UP (1996)
2. R.A.E.: Ortografía de la lengua española. Espasa Calpe, Madrid (1999)
3. Hammond, R.: The Sounds of Spanish: Analysis and Application (with Special Reference to American English). Somerville: Cascadilla Press (2001)
4. Navarro, T.: Manual de pronunciación española. Consejo Superior de Investigaciones Científicas, Instituto "Miguel de Cervantes", Madrid (1977)
5. Robinson, J.: Colors of poetry: Computational deconstruction. Master's thesis, Athens, Georga, USA, University of Georgia (2006)
6. Llorach, E.: Gramática de la lengua española. Espasa Calpe, Madrid (2003)
7. Miller, G.A.: Wordnet: a lexical database for English. Commun. ACM **38** (1995) 39–41
8. Rosenfeld, R.: Two decades of statistical language modeling: Where do we go from here? Proceedings of the IEEE **88** (2000)
9. Manurung, H.: An evolutionary algorithm approach to poetry generation. PhD thesis, School of Informatics, University of Edinburgh (2003)
10. Levy, R.P.: A computational model of poetic creativity with neural network as measure of adaptive fitness. In: Proccedings of the ICCBR-01 Workshop on Creative Systems. (2001)
11. Davies, M. (www.corpusdelespanol.org)
12. (http://www.gedlc.ulpgc.es/investigacion/desambigua/morfosintactico.htm)

Evolutionary GUIs for Sound Synthesis

James McDermott[1], Niall J.L. Griffith[1], and Michael O'Neill[2]

[1] Dept. Computer Science and Information Systems, University of Limerick
[2] NCRA, University College Dublin
jamesmichaelmcdermott@gmail.com, niall.griffith@ul.ie, m.oneill@ucd.ie

Abstract. This paper describes an experiment carried out to determine which, among several possible evolutionary and non-evolutionary sound synthesizer graphical user interfaces, is the most suitable for the task of matching a target sound. Results show that standard and new varieties of evolutionary interface are competitive with a standard non-evolutionary interface, achieving better results in some situations and worse in others. Subjects' comments suggest a preference for a new type of evolutionary interface, presented here, which allows faster audition of the population, avoiding the need for time-consuming fitness evaluation of poor-quality sounds.

1 Introduction

Evolutionary Computation (EC) has been applied by several authors to the problem of setting sound synthesizer parameters, using both automatic [1], [2], [3], [4], and interactive [5], [6], [7] EC methods. Little or no research has been reported on controlled experiments comparing Interactive EC (IEC) synthesizer GUIs (Graphical User Interfaces) with non-evolutionary synthesizer GUIs. This is one aim of this paper: the other is to introduce and study a novel IEC GUI using "sweeping".

Typically, software sound synthesizers are "played" by saved performance files or by MIDI instruments; these methods determine the choice of notes, their volumes and lengths, and sometimes a few other aspects of performance. Controlling these is usually a matter of learning to compose using a sequencer or to perform using a keyboard. Synthesizers also expose a number of continuously-variable *parameters* which affect the character of the emitted sound; controlling these does not require traditional virtuosity but does require understanding of their individual purposes in the synthesizer algorithms, a knowledge of the sound character being aimed for, and a great deal of persistence. This last fact provides the motivation for this work: IEC has the potential to make control of sound synthesis parameters much easier and more intuitive, and to remove the requirement for understanding of underlying synthesis algorithms.

1.1 Existing Work

IEC is EC driven by human evaluations of fitness: it can function as a way to avoid the problem of defining explicit fitness functions for hard-to-define goals; several authors have applied IEC to sound synthesis.

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 547–556, 2007.
© Springer-Verlag Berlin Heidelberg 2007

A typical IGA synthesis system is described by Johnson [6]. Here, genomes are floating-point arrays, which are translated into sounds by (i.e. serve as input to) the CSound *FOF* synthesizer. The user interface consists of buttons (to hear the sounds) and sliders (to assign them fitness values). After evaluating each generation, the user clicks an "evolve" button, causing the next generation to be created using mutation and crossover.

Genophone [7] is a complex interactive system, in which a dataglove, an evolutionary software interface, and a MIDI keyboard and synthesizer are used together. Evolution takes place at the levels of synthesis parameters and of performance parameters: thus the user awards highest fitness scores to individuals which produce desired sounds as well as desired performance mappings for the dataglove. The overall process is "exploratory rather than goal orientated; it is not designed to satisfy a priori sound specifications" [7].

MutaSynth [5] is an IGA application which can be applied to different synthesis engines via MIDI control, and also controls the evolution of score material. It has been used as a publicly-accessible installation, requiring that it be usable and controllable even by very casual users.

This research is successful in proving that IEC can be applied to sound synthesis, hiding the low-level details of synthesis from a user who prefers to think in aesthetic and perceptual terms. However there has been no attempt to quantitatively compare IEC interfaces with traditional ones.

Takagi writes that IEC applications suffer from a *fitness evaluation bottleneck* [8] – the fact that usually a human will be orders of magnitude slower than a computer in evaluating an individual's fitness. This means that the large population sizes and large numbers of generations, typical of automatically-driven EC, are infeasible – and so high-quality solutions are less likely to be found. Takagi discusses several ways in which researchers attempt to improve IEC systems: one of these is combination of interactive with non-interactive evolution. We have implemented new evolutionary interfaces which combine this technique with the idea of *sweeping*, or user-controlled interpolation.

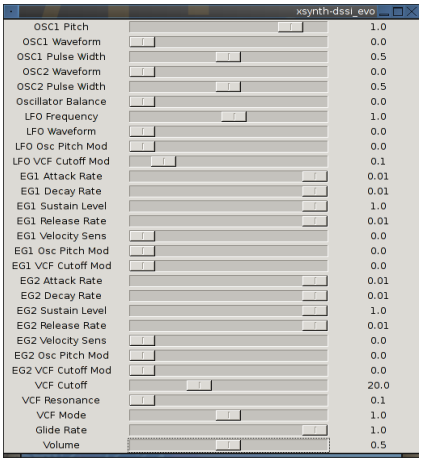
In the next section, we describe these ideas and the GUIs we have implemented and tested. In the following sections, we describe the protocol for the experiments, results and analysis, and finally conclusions and future work.

2 Software Implementation

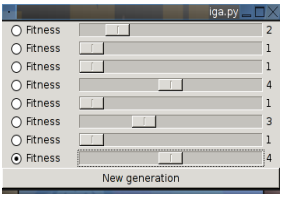
Our experiment compares four interfaces we have implemented for the Xsynth-DSSI synthesizer [9]:

GUI 0, "Sliders", is a standard non-evolutionary GUI (Fig. 1(a)), in which each parameter is set manually by the user using a slider. Each parameter's name and current value are displayed as text next to its slider. This GUI is intended as a control.

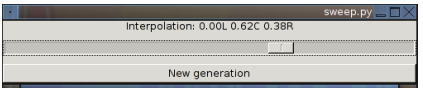
GUI 1, "IGA", is a plain IEC GUI (Fig. 1(b)), in which a population of sounds are available for evaluation. For each sound, the user sees a radio button and a slider. The radio button selects the given sound for playing, and the slider



(a) Non-evolutionary GUI



(b) Typical IGA GUI



(c) Sweeping IGA GUI

Fig. 1.

determines its user-awarded fitness (this is Takagi's "discrete fitness value input method" [8]). A "New Generation" button allows the user to declare that fitness evaluations are finished for this generation: at this point, a new generation is made using one-point crossover with probability 1.0 and Gaussian mutation of deviation width 0.75 with probability 0.3 (these values, higher than usual, were chosen on the basis of trial experiments in which increased diversity and variation between generations was found to be necessary). This GUI approximates one version of the state of the art in IEC sound synthesis applications, as used by Johnson [6]. Of the three existing systems mentioned in Sect. 1.1, only Johnson's is directly comparable with our goals, and sufficiently well-specified to allow re-implementation. It directly controls a synthesizer and nothing else. *Genophone* involves a dataglove and the learning of performance mappings, which are not present in our work. *MutaSynth* involves the evolution of both synthesis parameters and scores. Therefore, we have chosen Johnson's interface to be re-implemented as a control.

GUI 2, "Sweep", is a sweeping-style IEC GUI (Fig. 1(c)), in which a single slider is available to the user. This GUI is a contribution of this paper. Three discrete sounds L, C and R can be accessed by placing the slider at its leftmost, centre, and rightmost points. Sounds intermediate between L and C, and between C and R, can be accessed by moving the slider to intermediate points, as follows: when the slider is between L and C, at a distance x from L, the emitted sound X has parameters $X_i = L_i + x(C_i - L_i)$ (in fact, parameters are first mapped linearly or logarithmically, as appropriate, to the interval $[0, 1]$; then X is formed; finally it is mapped back to the true parameter space). The resulting sound can be thought of as a mixture of L and C proportional to the slider's nearness to those points. A similar arrangement holds for C and R.

Any pair of points close together on the slider will also be close in the parameter space, and will usually be quite similar in the sound space. Moving the slider thus usually results in quite a gradual change of the emitted sound over the majority of the slider's range (previous work [10] has supported this claim computationally, and it is also borne out by experience with the GUI). This arrangement allows the user to quickly audition a large number of individuals with a single mouse-gesture, focussing in on the most interesting areas, and wasting no time on awarding fitness values to poor-quality sounds.

A "New Generation" button allows the user to declare that a new generation is required: the current individual is retained as point C and new individuals are randomly generated for points L and R (alternatives including the generation of L and R by mutation from C were trialled in pilot experiments, but it was again found that greater diversity was required).

The "Sweep" operator thus allows the user to hand-control an interpolation at the genetic level between pairs of individuals. This does not violate the IEC principle that users should not need to understand the function of genes, since individual parameters are not exposed. However it certainly does violate any analogy with real-world evolution. It is comparable to the (non-interactive) *morphing* operator used in MutaSynth [5]. The evolutionary mechanism can also be compared with a $(1, 3)$ Evolutionary Strategy (ES) [11].

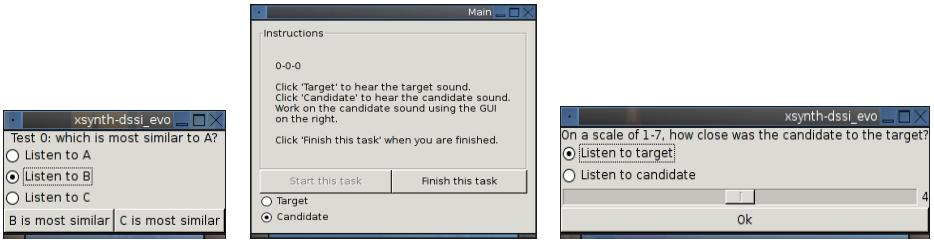
GUI 3, "Sweep with background evolution", is a sweeping-style IEC GUI (Fig. 1(c) again), augmented by background evolution. Here, a target waveform is loaded before any user interaction takes place. An automatic EC process then runs in the background, attempting to match the target waveform using a fitness function based on measurement of Attribute Distance between target and candidate sounds (see [10]). Meanwhile, the user interacts with the system as for GUI 2. In this case, the "New Generation" button indicates that the current individual is to be retained as point C; a new individual is to be randomly-generated for point L; and the best individual found so far by the background process is to be used for point R. It can happen that the user requests a new generation before the background process has found an improvement on the previous best individual: in this case a randomly-generated individual is used instead (in test experiments this is found to happen quite rarely). This is a type of Deme GA, in that at each generation one individual "migrates" from background to foreground. No migration in the opposite direction takes place.

The subjective impression of the first author is that this method does succeed in providing "raw materials" at point R which often are better than the randomly-generated sounds provided by GUI 2. It is useful in the real-world situation where a user already has a sound file exhibiting some desired characteristics, but wishes to re-synthesize to gain flexibility in pitch, loudness, duration, or timbre. It can be thought of as a way to "put knowledge in" to the system, and to exploit the complementary abilities of human and machine. However its use in this experiment is unrealistic in that here the target of background evolution is known to be exactly available using the synthesizer and is exactly the user's target.

3 Experimental Setup

3.1 Preliminary Discrimination Tests

Each user began by undertaking a series of 10 "triangle tests", in each of which the task was to listen to three sounds, A, B and C, and determine which of B and C was closest to A. In each triple of sounds, either B or C was in fact identical to A, while the other was a slightly mutated version of A. There was thus an objectively right answer to each triangle test. The purpose of this test was to gather data on how good subjects were in discriminating between sounds. The GUI for this test is shown in Fig. 2(a).



(a) Triangle Test GUI (b) Main Control GUI (c) User Rating GUI

Fig. 2.

3.2 Subjects

There were 20 subjects altogether, ranging in age between 23 and 45. There were 8 females and 12 males; in terms of synthesizer expertise, there were 6 beginners, 7 intermediate and 7 advanced, as classified by subjects themselves. Several were participants in or graduates of a one-year postgraduate level taught course in music technology.

Subjects were allowed to spend as long as they wanted, but in order to prevent them becoming fatigued or rushing the final experiments, they were advised to spend about 3-6 minutes on each task. The 8 tasks were presented in pairs, by GUI. Four different orderings of the pairs were used to avoid bias through learning or fatigue.

3.3 Target Sounds

Two target sounds were chosen from among the synthesizer's built-in preset sounds. This ensured that the target sounds were achievable using the given synthesizer, and were "desirable" sounds. They were: a percussive Xylophone-like sound (Target 0) and a typical "synth strings" sound (not a good imitation of real strings), with slow attack and release sections, and very slightly out-of-tune oscillators (Target 1). These sounds represent very different areas of the sound space.

3.4 Main Task

Figure 2(b) shows the GUI for the main part of the experiment. This GUI allowed subjects to start or finish the current task, and during the task, to switch the synthesizer back and forth between playing the target sound and the current candidate sound. In either case the sound was triggered every 1.5 seconds by a MIDI sequencer, and turned off 1 second later. On clicking the "Start this task" button, the GUI (0-3) for the current task was shown, allowing users to modify the candidate sound. On choosing to "Finish this task", subjects were presented with the User Rating GUI, shown in Fig. 2(c): here, they again listened to the target and candidate sounds, and awarded a score indicating how good a match had been achieved. After all tasks were finished, subjects filled in a short questionnaire.

4 Results and Analysis

A preliminary analysis of log files revealed that 4 subjects had, despite their instructions and trial period, failed to evolve through more than 0 generations in most of the evolutionary tasks (GUIs 1-3). This must firstly be taken as an indication that these simple GUIs are not as obvious in their function as intended. For statistical analysis, the data generated by these 4 subjects was eliminated from the dataset.

Subjects' scores for the triangle tests were all very high: it seems to have turned out to be much easier than intended (and easier than expected, on the basis of a short pilot experiment). Every subject scored either 9 or 10 out of 10. This indicates that at least at the beginning of the experiment, subjects were taking the experiment seriously and attempting to give the correct answers, and it indicates that all subjects were capable of discriminating between different sounds. The purpose of the triangle tests was to differentiate among subjects according to their ability in this regard, but since all subjects were so successful, this analysis can not be carried out.

For each task, the Subject ID, GUI, Target, Task Ordering, Time Taken and User Rating were recorded. The distance between the target sound and the sound produced by the subject was measured in three ways:

Attribute Distance works by extracting a set of 40 timbral, perceptual, and statistical attributes [12] from the sounds (via digital signal processing methods). Each attribute is then mapped, linearly or logarithmically as appropriate for perceptual reasons, from its true range to the range $[0, 1]$. The distance between two values for an attribute is then the absolute value of their difference, and an overall distance between a pair of sounds is defined by averaging these individual distances.

DFT Comparison works by dividing each sound into overlapping windows, for each window taking the Discrete Fourier Transform, and finally comparing the power in corresponding transform bins. This method is the most commonly-used in the EC synthesis literature.

Parameter Distance compares not sounds but the parameter settings which give rise to them. Each parameter is mapped from its true range to the interval $[0, 1]$, again either linearly or logarithmically as appropriate. Two sounds are compared by taking the average of the absolute value of the differences of individual mapped parameters.

4.1 Main Results

Table 1 shows the results of a two-way repeated measures ANOVA for each of User Rating, Time Taken, and Attribute Distance, where the two factors are GUI and Target.

Table 1. Average results for User Rating (user's satisfaction with the match, on a scale of 1-7; higher is better), Time Taken (in seconds; lower is better) and Attribute Distance (from target sound to achieved sound, on a unit scale; lower is better), analysed by GUI and by target sound

		GUI 0	GUI 1	GUI 2	GUI 3	All GUIs
User Rating (1-7)	Target 0	3.44	3.06	2.50	3.06	3.02
	Target 1	3.87	4.12	4.50	4.56	4.27
	Both Targets	3.66	3.59	3.50	3.81	3.64
Time Taken (s)	Target 0	475	455	271	227	357
	Target 1	273	292	116	135	204
	Both Targets	374	374	194	181	281
Attribute Distance	Target 0	0.14	0.18	0.18	0.17	0.17
	Target 1	0.08	0.13	0.09	0.07	0.09
	Both Targets	0.11	0.16	0.13	0.12	0.13

There are no statistically significant differences in User Rating between GUIs ($F = 0.38$; $p > 0.1$). However, there are significant differences by Target ($F = 34.8$; $p < 0.001$) and by GUI against Target ($F = 3.14$; $p < 0.05$). In particular the Slider GUI (0) received the best ratings for Target 0, but the two Sweep GUIs (2 and 3) received the best ratings for Target 1. This result is reflected in the statements by two expert synthesizer users that the Sweep interfaces were more appropriate for timbral matching (the main difficulty in Target 1), while the standard GUI (0) was more appropriate for envelope matching (the main difficulty in Target 0). Thus there is evidence that the Sweep interfaces are useful in particular circumstances.

Users spent significantly less time working with the Sweep GUIs (2 and 3) than the other two ($F = 12.1$; $p < 0.001$). This is a positive result, since achieving the same quality of match in a shorter time benefits both the serious and the casual user.

There are significant differences in Attribute Distance by GUI ($F = 10.8$; $p < 0.001$): the Sliders interface (0) gave the lowest Attribute Distances (performs

best), while the IGA (1) was worst: the two Sweep interfaces were in between. Again, these results are differentiated strongly by target: for target 0, Sliders was best and all others equally bad; for target 1, IGA was worst and all others about equally good. There were no significant differences in User Rating by Task Ordering ($F = 1.24$; $p > 0.1$).

One of our motivating hypotheses was that the Sweep interfaces would be more suitable for novices. Although this hypothesis is partly confirmed by users' responses to the questionnaire, results show that there were no statistically significant differences in the User Ratings by GUI and Synthesizer Experience Group ($F = 1.40$; $p > 0.1$).

The results show significant differences in User Rating ($F = 34.8$; $p < 0.001$), Attribute Distance ($F = 217.0$; $p < 0.001$), and Time ($F = 12.1$; $p < 0.001$), by Target. This indicates that one target is more difficult than the other. This can be confirmed using the method of random sampling: we have generated 100 sounds at random in the parameter space and for each, calculated its Attribute Distance to each of the two targets. The distance to target 0 was found to be greater ($p < 0.001$): the mean Attribute Distances were 0.185 and 0.140 respectively. A random point in the Parameter space is likely to be closer to target 1. This implies that the map from Parameter to Attribute space is denser in the area of target 1 than in that of target 0, so in a sense target 0 is more difficult to find. It is likely that variations in the density of this map are characteristic of the synthesizer; this is a key issue in the problem of target-matching. Future work could use this insight in to guide the generation of new individuals in interactive algorithms.

4.2 Qualitative Analysis

In the post-test questionnaire, subjects were invited to give any comments they wished on the GUIs and experiments. Of the 9 subjects who chose to express a preference, six said that one of the Sweep GUIs was the best. One subject remarked "There was a sense of progression with the sweep GUI, whereas the sliders GUI didn't have that."

Two advanced synth users remarked that individual parameters (i.e. a standard interface) were best for setting time envelopes, while the Sweep interface was best for setting timbral aspects of the sound. This seems to be partly due to expert users finding that the evolutionary algorithms (GUIs 1-3) did not provide the necessary variation in time envelope "raw material". One further remarked that it was possible to form a mental picture of the sound's time envelope, and then to match this against a representation of the desired sound's time envelope, but that timbre resisted this mental representation.

Several users asked for features to be added, and in particular a "back" button or a save facility, in the case of the evolutionary interfaces. (These were considered during the design phase, but rejected as imposing too much interface complexity on subjects.) Often the same subjects expressed frustration that they had achieved quite a good match only to lose it in the next generation. This

applies particularly, but not exclusively, to the IGA GUI (1), since the Sweep GUI retains the "best" sound from the previous generation. One user also asked for a "reset to default" button, in the case of the standard synth interface.

In response to the potential objection that novices could quickly be taught to become intermediate synthesizer users, and thus proficient in the use of a typical synthesizer interface, we note that of the several subjects who were graduates of at least a one-year graduate-level course in music technology, some still regarded themselves as no more than intermediate-level synthesizer users and in some cases remarked that the synthesizer parameters were confusing or that they didn't understand what they did.

4.3 Correlations Among Measures of Success

Table 2 gives the Pearson's product-moment correlation (and associated 95% confidence interval) between User Rating and several other measures of performance. Parameter Distance and Attribute Distance are quite strongly negatively correlated with User Rating. This lends some support to the use of Attribute Distance as a fitness function for automatic evolution, and to the use of Parameter Distance as a measure of evolutionary success for experimental use.

Table 2. Correlations between User Rating and other measures of success

Measure	Pearson's correlation	Confidence interval	Significance
Parameter Distance	-0.32	[-0.47, -0.16]	*
Attribute Distance	-0.51	[-0.63, -0.37]	*
DFT Distance	0.17	[-0.00, 0.33]	
Time	-0.18	[-0.34, -0.01]	

5 Conclusions and Future Work

We have introduced and studied a novel IEC GUI using the technique of "sweeping". Overall, the Sweep interfaces with and without background evolution have been shown to be competitive with and in some ways better than the other interfaces, as judged by User Ratings, Attribute Distances, and Time spent per task.

Although the technique of background evolution has not been shown to provide a statistically significant improvement in performance, each of User Ratings, Times, and Attribute Distances are in almost every category slightly better for GUI 3 than GUI 2. As noted above, GUI 3 represents an idealised situation: for these experiments, the target is exactly available using the given synthesizer, and a perfect recording of the target is available. Real-world use would further diminish any improvement due to background evolution.

The correlations between Attribute Distance and User Rating, and between Parameter Distance and User Rating, have application to the design and testing of automatic search algorithms.

Several modifications are suggested by the data and by user comments: extra features such as a "back" button or a save facility; modifications to the method of choosing the endpoints for the Sweep interface; and the extension of the Sweep interface to the 2-dimensional case.

Acknowledgements

The first co-author gratefully acknowledges the guidance of his co-authors and supervisors; many thanks also to Brian Sullivan and Dr. Jean Saunders, Statistics Consulting Unit, University of Limerick, and Dr. Fred Cummins, University College Dublin. The first co-author is supported by IRCSET grant no. RS/2003/68.

References

1. Horner, A., Beauchamp, J., Haken, L.: Machine tongues XVI: Genetic algorithms and their application to FM matching synthesis. Computer Music Journal **17**(4) (1993) 17–29
2. Garcia, R.A.: Growing sound synthesizers using evolutionary methods. In Bilotta, E., Miranda, E.R., Pantano, P., Todd, P., eds.: Proc. ALMMA 2001: Artificial Life Models for Musical Applications Workshop (ECAL). (2001)
3. Mitchell, T.J., Sullivan, J.C.W.: Frequency modulation tone matching using a fuzzy clustering evolution strategy. In: Audio Engineering Society 118th Convention. (2005)
4. Riionheimo, J., Välimäki, V.: Parameter estimation of a plucked string synthesis model using a genetic algorithm with perceptual fitness calculation. EURASIP Journal on Applied Signal Processing **8** (2003) 791–805
5. Dahlstedt, P.: A MutaSynth in parameter space: interactive composition through evolution. Organised Sound **6**(2) (2001) 121–124
6. Johnson, C.G.: Exploring sound-space with interactive genetic algorithms. Leonardo **36**(1) (2003) 51–54
7. Mandelis, J., Husbands, P.: Genophone: Evolving sounds and integral performance parameter mappings. In: EvoWorkshops. (2003)
8. Takagi, H.: Interactive evolutionary computation: Fusion of the capabilities of EC optimization and human evaluation. Proc. of the IEEE **89**(9) (2001) 1275–1296
9. Bolton, S.: XSynth-DSSI (2005) `http://dssi.sourceforge.net`, last viewed 2 March 2006.
10. McDermott, J., Griffith, N.J.L., O'Neill, M.: Evolutionary Computation Applied to Sound Synthesis. In: The Art of Artificial Evolution. Springer (2006)
11. Beyer, H.G.: The Theory of Evolution Strategies. Springer (2001)
12. McDermott, J., Griffith, N.J.L., O'Neill, M.: Timbral, perceptual, and statistical attributes for synthesized sound. In: Proc. of the International Computer Music Conference. (2006)

Evolving Music Generation with SOM-Fitness Genetic Programming

Somnuk Phon-Amnuaisuk, Edwin Hui Hean Law, and Ho Chin Kuan

Music Informatics Research Group,
Faculty of Information Technology, Multimedia University,
Jln Multimedia, 63100 Cyberjaya, Selangor Darul Ehsan, Malaysia
somnuk.amnuaisuk@mmu.edu.my, edwin@mmu.edu.my, ckho@mmu.edu.my

Abstract. Most real life applications have huge search spaces. Evolutionary Computation provides an advantage in the form of parallel explorations of many parts of the search space. In this report, Genetic Programming is the technique we used to search for good melodic fragments. It is generally accepted that knowledge is a crucial factor to guide search. Here, we show that SOM can be used to facilitate the encoding of domain knowledge into the system. The SOM was trained with music of desired quality and was used as fitness functions. In this work, we are not interested in music with complex rules but with simple music employed in computer games. We argue that this technique provides a flexible and adaptive means to capture the domain knowledge in the system.

Keywords: Genetic Programming, Self-Organising Features Map, Automatic Music generation.

1 Background

Genetic Programming (GP) [8] is one of the search techniques under evolutionary computation. It is well-accepted in literature that the application of heuristics to guide search is the most effective tactic in exploring an intractable search space. Unlike a conventional single agent search, GPs may be seen as performing parallel heuristic search. Domain knowledge and heuristic are encoded in a GP engine in the form of (i) representations of individuals, (ii) how the individual is actually altered and (iii) the selection and reproduction procedure of that GP. The problem set-up, the representation scheme and the evolutionary mechanism are important factors which help to contribute to the success of the application.

One of the most influential factors in an evolutionary mechanism is the fitness functions. There are three main common approaches to obtain the fitness functions: (i) from the domain expert, (ii) from interactions with users and (iii) from examples. Building fitness functions from the domain knowledge is the most common tactic since it is natural in most applications that the representation of good individuals and bad individuals (in the population) can be explicitly stated. In some domain (especially in art), it could be quite hard to explicitly code the fitness functions that some researchers have resorted to interactive

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 557–566, 2007.
© Springer-Verlag Berlin Heidelberg 2007

feedback from users. The interactive feedback approach is time-consuming and the feedback may lack consistency. The last approach (building fitness functions from examples) has been explored and has gained more interest recently. In this approach, examples of desired qualities could be used to determine the fitness functions dynamically. Hence it is flexible and also adaptive.

This paper discusses our application of Genetic Programming and Self-Organising Map (SOM) in a music-generation task. In our work, the GP generated melodic fragments and the SOM was used to provide fitness measurements of these melodic fragments. We organise the presentation into the following topics: (i) Background, (ii) Problem statement, (iii) Literature review, (iv) GP with SOM critic, (v) Result and (vi) Conclusion.

2 Problem Statements

The fitness functions play important roles in evolutionary computing. Constructing fitness functions as rules in a traditional GP system is time- consuming. In this paper, we propose a simple yet effective way of expressing fitness functions using self-organising features map (SOM) [7]. In this approach, the SOM

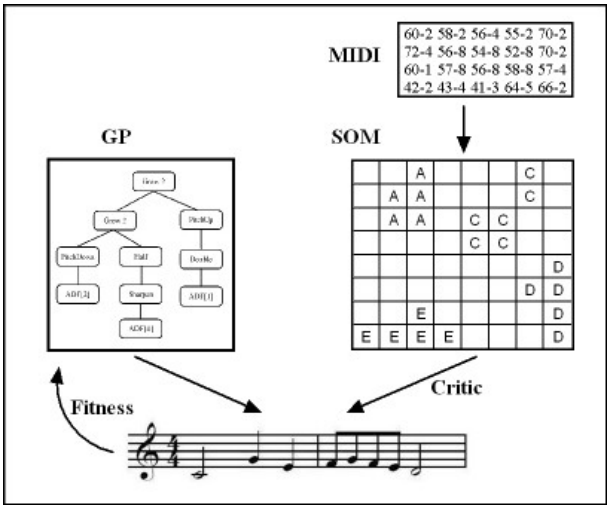


Fig. 1. GP with SOM critic

is trained with the desired music and the evaluation is based on the similarities of the evolved music and the desired qualities. The desired musical qualities are crucial here. Depending on the set-up of these similarity-measurement criteria, the output of the system could be varied from imitation to inspiration. Figure 1 illustrates the main idea of this work. The SOM is trained with the desired music

quality and the SOM is used as a critic to the GP output. We argue that this approach offers a flexible and adaptive means to encode domain knowledge as fitness functions.

3 Literature Review

Automatic music generation can be set up as an optimisation problem. Optimisation using traditional search techniques (e.g. graph-search or tree-search) or computational models such as ANNs traverse the search space with the guide of knowledge coded in terms of heuristic (in most search techniques) or in terms of learned search-space landscape. The main distinctive feature of these optimisation techniques is that the search space is explored in sequence through state transformations using heuristic or gradient information. In contrast to the above traditional optimisation approach, [5], [10], *Evolutionary Computation* (EC includes GAs, GPs, ACOs) techniques explore many parts of the search space at once. Local and global explorations could be tuned to enhance the quality of the solution. This proves to be useful in solving many problems. In the music domain, [1], [6] are among the early works in this area. More detailed overviews of the previous works could be found in [2] and [14].

To be critical, the quality of the music generated from EC cannot be compared to human composers. It is now appreciated that the *"no free lunch theory"* is generally true. It is unlikely that EC would magically produce a genius piece of work without sufficient a priori knowledge. The issue, therefore, is what is a better way to exploit the knowledge (e.g. representation of individual, reproduction scheme, fitness function) in the EC paradigm (see also [4], [9]).

Fitness evaluation is one of the most important components in EC. It directly determines the quality of the output. The fitness functions could be stationary (i.e. do not change with time) or dynamic; Local (i.e. evaluation is performed at a local level) or global; or the combination of both. Coding the above characteristics of the domain knowledge as fitness functions is not a simple task. The task demands a good understanding of the domain knowledge. If we want a good performance then we must have more knowledge in the system. Hence, flexible and adaptive knowledge encoding technique is an important and useful research issue.

In recent years, many researchers have experimented with SOM as a means to capture domain knowledge. Encoding knowledge using SOM is useful in displaying the relationships among musical concepts. These concepts could be tonality, rhythmic patterns, melodic contour, FFT coefficients, etc. The encoded concepts depend on the choices of representation and its application. Recently, music information retrieval is one of the main interests and SOM has been used to cluster melodic structural patterns of music in various styles (e.g. classical period, romantic period and Jazz [11], popular music [12] and folk music [15]).

Our work focuses on generating a melody using GP and SOM. GP acts as a creative composer and SOM acts as a critic. Our choice of representation

is abstraction at a pitch-time level. More details of the system and its fitness evaluation will be discussed in the next section.

4 GP with SOM Critic

Genetic Programming. A distinctive feature of evolutionary approach is that it performs parallel searches on the search space. Each individual may be seen as a state in the search space. The local or global characteristics of the parallel search are determined through the genetic operators.

Self Organising Features Map. Self-Organising Features Map (SOM) [7] is a useful technique for mapping the data in a higher-dimension space to the data in a lower-dimension space (in our work, this lower dimensional space is in the form of a 2D map). The SOM algorithm projects input vector x in R^n into a 2D array map of i neurons, that is, each of the neuron i receives an input from the input vector $x \in R^n$. Each neuron has a model vector (weight vector) $m_i \in R^n$ associated with it.

Similar input patterns group themselves forming a cluster in the map. The forming of clusters are based on the so-called "the winner takes all" strategy. The winner c is defined by:

$$\|x(t) - m_c(t)\| \leq \|x(t) - m_i(t)\| \text{ for all } i.$$

The regression of the model vector is performed using the following update rule:

$$m_i(t+1) = m_i(t) + \eta(t) h_{c,i}(t)(x(t) - m_i(t)),$$

where η is the learning rate and $h_{c,i}(t)$ is the neighborhood function for smoothing neighboring neurons of the winner.

4.1 Knowledge Representation

Since conceptualization of music at a coarse grain level is natural, representing musical concepts at a higher abstraction level is universal in most works (e.g. music is realised in terms of pitch, duration, interval, harmony, dynamic and so on). The representation of the computational model in the shape of production rules $A_1 \wedge A_2 \wedge \ldots \wedge A_n \rightarrow B$ using *logic* [3], [16]; conditional probability $P(B|A_1, A_2, ..., A_n)$ using *probabilistic* and *statistical* models are common [13]. Looking at music at the lower grain (e.g. in terms of its frequency content) is useful in many applications such as in a synthesis task. In recent years, employing features obtained from signal-processing techniques in music research has gained more attention, especially in music information retrieval. However, this approach has a serious drawback in the mapping between high level semantic of musical structures (e.g. C major) and low level features (e.g. FFT coefficients). In music composition, it is much more natural to work with the higher level of conceptual abstraction (e.g. pitch, duration, interval, etc.). This is mainly why the representation of our work is at the high-level abstraction.

Representation of Each Individual GP. In this implementation, each GP individual is a *strongly typed GP tree* that represents a melody line. The function set consists of two classes of functions: the *branching functions* and the *musical functions*. The branching functions occur at the top of the tree and act to concatenate individual notes to form a melody by taking two or four inputs. The musical functions occur below these and they manipulate the pitch and duration of each note where each function takes and returns one note. Figure 2 shows an

Nodes	Description
Grow2/Grow4	Takes in 2 or 4 inputs and returns the left-to-right concatenation of the input notes
PitchUp/PitchDown	Increases or decreases the pitch of the note by one tone.
Sharpen/Flatten	Adds a Sharp or a Flat to the note.
Double/Half	Doubles or halves the duration of the note.
ADFs	Automatic Defined Functions e.g. ADF[2] refers to function number two

Fig. 2. Example of functions

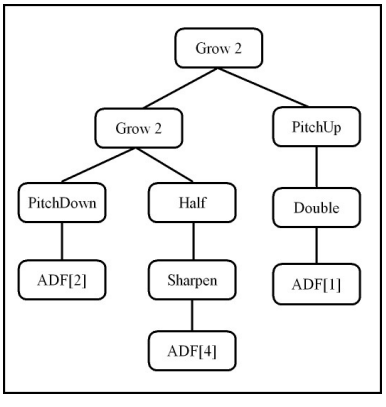


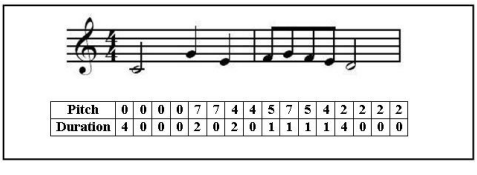
Fig. 3. An individual GP

example of node types in our implementation and Figure 3 shows an example of each individual GP chromosome. The terminal set always consists of notes represented in the form of *ADFs (Automatic Defined Functions)*. Each ADF is a tree with one root node and 4 branches, with the node representing a note and each branch representing the pitch, duration, octave and accidental property of the note using *ERCs (Ephemeral Random Constants)*. The ADF structure is fixed and each individual has 5 root notes (ADFs) to form the melody with. *PitchERC* holds a value of 1 to 7 representing pitches C to B. *DurationERC* holds an integer value where a value of 2, 4, or 8 corresponds to a minim, a crotchet and a quaver respectively. *OctaveERC* holds an integer value where 0

represents the octave of middle C, -1 and +1 represent a decrement and increment in the octave range respectively. Finally, *AccidentalERC* holds a value of -1,0 or 1 for a flat, natural or sharp respectively.

SOM Input Vector Representation. For the SOM fitness raters, we represent a melody (input vector of the SOM) in a relative pitch based on the original MIDI form (this is convenient since the input music is in the MIDI format). Each MIDI pitch value is converted into a relative representation by calculating the difference between it and the value of the input musics native key. Input vectors of size thirty-two (32) are used for the SOM and each of these inputs represents a four-bar melody in 4/4 time (note that Figure 4 shows an example of a two-bar input vector).

For each input, a value is stored for the pitch and duration of the note that is playing at that particular segment. The pitch (i.e. number of semitones from the native key of the tune) and length (in terms of number of quaver beats) of the note is stored at the starting point of the note, while at each proceeding point where the note is being held, a value of 0 is assigned for the duration. A pitch value of 99 (a number will not likely occur in the relative pitch) is assigned for rests (see Figure 4). With this scheme, both the rhythm and tempo of a melody



Pitch	0	0	0	0	7	7	4	4	5	7	5	4	2	2	2	2
Duration	4	0	0	0	2	0	2	0	1	1	1	1	4	0	0	0

Fig. 4. An input vector for SOM

are encoded at each node in the SOM. Further information such as the contour and harmony of the melody can also be extracted from each node by analyzing the contents of the melody.

4.2 Automatic Compositions Process

Experiments for GP were done in ECJ[1], and we utilized mostly the parameters recommended by Koza. Our population contained 1000 individuals and they were bred for 50 generations. The initial population during generation 0 was seeded using the *FullBuilder algorithm* described by Koza with a fixed tree-depth of 15. Breeding was done using Koza Crossover and Mutation with the probabilities of 0.8 and 0.2 respectively. Individuals selected for breeding were chosen using Tournament Selection with a selection pool size 7.

The SOM we used was a square map of size 20×20. For each training run, the SOM was trained with a starting learning rate of $\alpha = 0.03$, and training proceeded for 300 iterations (empirical decision). The learning rate (η) and distance

[1] Available from http://cs.gmu.edu/ eclab/projects/ecj/

fall-off rate (h) are adaptive. Their values for every iteration (i) are calculated based on the following formulae:

$\eta = \alpha e^{-\frac{i}{\beta}}$: where β is the maximum iteration.
$h = e^{-\frac{\delta^2}{2R^2}}$: where δ is the distance on the map and R is the neighborhood radius related to the size of the SOM.

At the beginning of each GP run, the SOM is trained with a set of input melodies to form the basis for the fitness rating of that run. During this training, weight adjustments are done in a 2 pass process using the formulae shown above. In the first pass, the duration vector is adjusted based on the location of the downbeats (start of each note), the location of the downbeats, and its value (length of the note) are both adjusted. In the second pass, the pitch values are then adjusted based on the new locations of the downbeats, only the pitches at the downbeats are adjusted while proceeding points where the note is being held will be copied with the new downbeat pitch value.

At each generation, each of the 1000 GP individuals will be mapped onto the trained SOM. This mapping is done by evaluating each GP individual similarities to the SOM contents in terms of its Tempo, Pitch and Harmony (see Figure 5). For the Tempo and Pitch metrics, the differences of the values are measured at each one of the corresponding 32 points of the SOM vector. To allow for this, the generated GP individuals pitch vector has to be converted to relative values by subtracting each pitch value with the pitch value of its first note (we assume the first note is the tunes native key). For harmony, the frequency of occurrences of the pitches that appear in each bar is measured. This data is then used to measure the harmony of the GP individual, where a match with a frequently occurring pitch produces high fitness vice versa.

Parameters	Measure similarity metrics from
Tempo	Duration values at each of the 32 points.
Pitch	Pitch values at each of the 32 points.
Harmony	Membership of pitch for the implied harmony in each bar.

Fig. 5. Multi-criteria fitness metric

5 Experiments and Discussion

Figure 6 illustrates the effects of adjusting different evaluation measures. In this example, a famous *Twinkle, Twinkle, Little star* melody was used to train the SOM. The GP was then run multiple times (400 runs, equivalent to 1600 crotchet notes). A different weight was assigned to the similarity measurement that put more emphasis on Pitch, Tempo, Harmony and equal emphasis on all (100 runs each, the similarity measurements are then averaged). These experiments demonstrated the effects of each of these properties to the output of the system.

	Normalised similarity		
	Harmony	Pitch	Tempo
Emp Harmony	1.00	0.20	0.95
Emp Pitch	1.00	0.70	0.85
Emp Tempo	0.35	0.10	1.00
Emp All	1.00	0.55	0.98
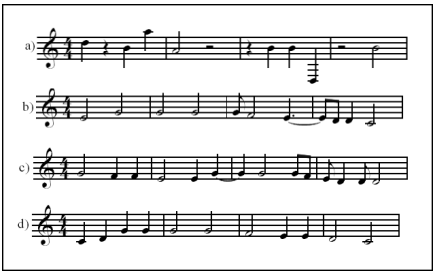


Fig. 6. Output from the Twinkle, Twinkle little star training examples. (a) Output using Tempo metric. (b) Output using Pitch metric. (c) Output using Harmony metric. (d) Output from combination of Tempo, Pitch and Harmony metrics.

When using each of the similarity measurement individually to evaluate the GP's individuals, the output produced was within expectation. Tempo produced output that mimicked the constant rhythm of the input music but with wild pitch fluctuations, Pitch produced output that showed similar progressions as original input while the notes in each bar of the output evaluated using Harmony were restricted to those available in the corresponding input bar.

5.1 Can We Produce Variations from a Given Theme?

Figure 8 is the output generated from fragments of input taken from *Super Mario Bros*, a classic game tune. The training input is shown in Figure 7.
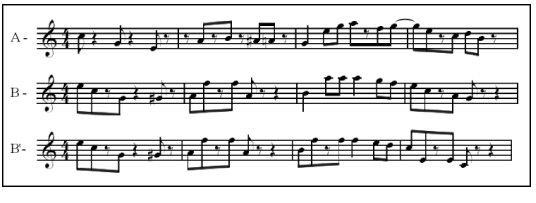


Fig. 7. Training examples of A, B and B'

There are two important questions we need to answer in the above experiment. (i) What are the suitable metrics to be used (e.g. pitch, harmony, etc)? (ii) What are suitable measurement of those metrics? (i.e. how do we measure similarity in harmony?) We do not have clear answers to these questions yet. In general, suitable metrics should capture the essentials of the style for a particular genre and suitable measurements of those metrics should produce reasonable output. We believe this is an interesting issue to be explored for many reasons. Traditional knowledge acquisition from domain experts is time-consuming. In our proposed paradigm, the domain knowledge could be acquired with ease and knowledge maintenance can be done by re-training the SOM. However, it is hard
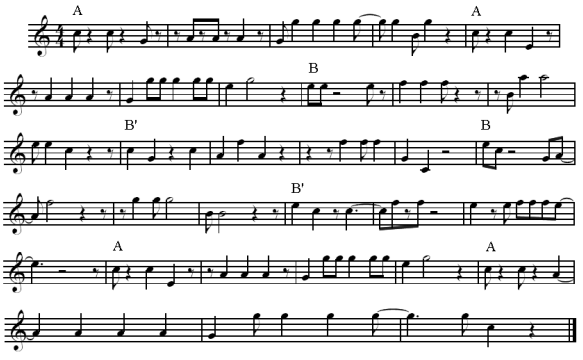


Fig. 8. Output from the system (with GP and SOM critic)

to see whether the current proposed metrics are enough for generating interesting music. The issue of dependency between these metrics also play a crucial role in this problem. More structured knowledge is needed.

6 Conclusion

Coding domain knowledge is always an issue in music. With too much knowledge, the system cannot produce good music as it lacks creativity. On the other hand, with too little knowledge, the system produces nonsensical noise as it lacks sensibility. In this report, we present a composition system that relies on two techniques: GP and SOM. GP acts as a very creative composer and SOM as a critic. In this experiment, we do not load the GP with domain knowledge and we code the SOM with music examples. In this paradigm, SOM provides a flexible and easy means to capture the domain knowledge and GP explores the search space around the given examples. In our experiment, we exploited three different evaluation criteria: contour, duration and harmony. Musical rules were loosely captured from music examples used in training the SOM. The creativity (if we are allowed to say that) was from the weight of different evaluation functions. The experiment showed very encouraging results.

References

1. Biles, J. A.: GenJam: a genetic algorithm for generating jazz solos. In *Proceedings of the International Computer Music Conference*, 1994. pages 131-137. Aarhus, Denmark.
2. Burton, A.R. and Vladimirava, T.: Generation of musical sequences with genetic techniques. *Computer Music Journal*, 23(4):59-73, 1999.
3. Courtot, F.: Logical Representation and Induction for Computer Assisted Composition. In Balaban, M., Ebcioglu, K. and Laske, O., editors, *Understanding Music with AI: Perspectives on music cognition*, chapter 7, pages 157-181. The AAAI Press/The MIT Press, 1992.

4. Gartland-Jones, A. and Copley, P.: The suitability of genetic algorithms for musical composition *Contemporary Music Review*, 22(3):43-55, 2003.
5. Ebcioglu, K.: An expert system for harmonizing four-part chorales. In M. Balaban, K. Ebcioglu, and O. Laske, editors, *Understanding Music with AI: Perspectives on music cognition*, Chapter 12 pages 294-333. The AAAI Press/The MIT Press.
6. Horner, A. and Goldberg, D. E.: Genetic algorithms and computer-assisted music composition. In R. Belew and L. Booker, editors, *The Fourth International Conference on Genetic Algorithms, 1991, Proceedings*, San Francisco, CA: Morgan Kauffman.
7. Kohonen, T.: *Self-organising Maps (2nd ed.)*. Berlin: Springer-Verlag, 1997.
8. Koza, J. R.: *Genetic Programming: On Programming of Computers by Means of Natural Selection*. The MIT Press, 1992.
9. Miranda, E. R.: On the music of emergent behaviour: what can evolutionary computation bring to musician? *Leodarno*, 36(1), 55-88, 2003.
10. Phon-Amnuaisuk, S.: Control language for harmonisation process. In Christina Anagnostopoulou, Miguel Ferrand, and Alan Smaill, editors, *Music and Artificial Intelligence, Second International Conference, ICMAI 2002, Edinburgh, Scotland, UK, September 12-14, 2002, Proceedings*, volume 2445 of *Lecture Notes in Computer Science*. Springer, 2002.
11. Ponce de León, P. J. and Inesta, J. M.: Musical style classification from symbolic data: A two-styles case study. In U. K. Wiil, editor, *Computer Music Modeling and Retrieval, International Symposium, CMMR 2003,Montpellier, France, May 26-27, 2003, Revised Papers*, volume 2771 of *Lecture Notes in Computer Science*. pp. 166-177. Springer, 2004.
12. Skovenborg, E. and Arnspang, J.: Extraction of structural patterns in popular melodies. In U. K. Wiil, editor, *Computer Music Modeling and Retrieval, International Symposium, CMMR 2003,Montpellier, France, May 26-27, 2003, Revised Papers*, volume 2771 of *Lecture Notes in Computer Science*. pp. 98-113. Springer, 2004.
13. Temperley, D.: *The Cognition of Basic Musical Structure*. The MIT Press, 2001.
14. Todd, P. M. and Werner, G. M.: Frankensteinian methods for evolutionary music composition. In N. Griffith and P. M. Todd, editors, *Musical Networks: Parallel Distributed Perception and Performance*, pages 313-340. The MIT Press.
15. Toiviainen, P. and Eerola, T. A method for comparative analysis of folk music based on musical feature extraction and neural networks. In *VII International Symposium on Systematic and Comparative Musicology and III International Conference on Cognitive Musicology*, University of Jyvskyl, Finland, August 16-19 2001.
16. West, R., Howell, P. and Cross, I.: Musical structure and knowledge representation. In Howell, P., West, R. and Cross, I. editors, *Representing Musical Structure*, chapter 1, pages 1-30. Academic Press, 1991.

An Automated Music Improviser Using a Genetic Algorithm Driven Synthesis Engine

Matthew John Yee-King

Creative Systems Lab, Department of Informatics, University of Sussex, Brighton, UK

Abstract. This paper describes an automated computer improviser which attempts to follow and improvise against the frequencies and timbres found in an incoming audio stream. The improviser is controlled by an ever changing set of sequences which are generated by analysing the incoming audio stream (which may be a feed from a live musician) for its physical and musical properties such as pitch and amplitude. Control data from these sequences is passed to the synthesis engine where it is used to configure sonic events. These sonic events are generated using sound synthesis algorithms designed by an unsupervised genetic algorithm where the fitness function compares snapshots of the incoming audio to snapshots of the audio output of the evolving synthesizers in the spectral domain in order to drive the population to match the incoming sounds. The sound generating performance system and sound designing evolutionary system operate in real time in parallel to produce an interactive stream of synthesised sound. An overview of related systems is provided, this system is described then some preliminary results are presented.

1 Introduction

The construction of complete, automated improvisation systems which respond in real time to the sounds made by human improviser(s) by generating complementary sonic gestures is a well established activity in computer music research. ([1],[2], [3]) Typically these systems comprise the following parts:

1. An analysis system which extracts features from the sound being made by the human musician such as pitch and timbre.
2. A storage system which remembers the analysis data, e.g. storing a list of the notes that the human musician has played.
3. A playback system which interprets the stored analysis data into sound, e.g. playing back notes previously played by the musician using some sort of synthesis engine.

This paper presents preliminary work on such a system, which interprets an audio stream from a live musician into control data for an ever changing synthesis engine which is attempting to match timbres found in the audio stream. The main research themes in this work are interactive GAs for sound design, unsupervised GAs for sound design, live algorithmic music (this term after [3])and computational creativity. The motivation for this system is partly creative and

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 567–576, 2007.
© Springer-Verlag Berlin Heidelberg 2007

partly technical, being an attempt to create an interesting computer improviser as well as an investigation of the use of unsupervised genetic algorithms in a real-time context.

The paper comprises 5 sections - this introduction, related work, a technical description of the system, some initial results and the future plans for this work.

2 Related Work

Several researchers have used interactive genetic algorithms to allow a user to guide the evolution of parameters for a sound synthesis algorithm. Such systems allow even a novice 'sound synthesist' to explore a complex space of possible parameters to find unique and dynamic sounds, sounds which they would not otherwise have been able to design, [4,5]. A limitation of such systems is the user's inability to audition large numbers of different sounds. In [6], a collaborative, networked interactive GA is described which pools the knowledge of several users in an attempt to solve this problem.

Another solution is to allow the user to somehow specify a target sound or timbre and use an unsupervised GA to search for a synthesis algorithm which can generate a sound resembling the target. Such systems are closer to Holland's classic GA [7]. In [8], Mitchell et al evolved parameters for an FM synthesis algorithm, after [9,10] which would cause it to generate a sound similar to that made by a random set of parameters applied to the same algorithm. The fitness function took several snapshots of the candidate sound in the frequency domain and compared these to snapshots of the target sound. The main limitation of this system is that is does not attempt to match timbres of arbitrary complexity, rather it attempts to match timbres generated by the same synthesis algorithm used by the evolving sounds, a simpler search space. Also it limits the evolved structures to parametric permutations of a fixed synthesis algorithm. Whilst FM synthesis can generate a fairly wide range of timbres, it is felt that GAs are more creatively exciting when allowed to design their own structures rather than eliciting parameters for a fixed structure.

In [11] a GA was used to adjust the parameters of a synthesis algorithm to match a timbre description in the form of levels of different timbral properties such as brightness and warmth. The evolving sounds were assessed by feeding measures of their most prominent partials and their amplitude envelope into a feed forward neural network trained using the back propagation algorithm to match these sonic parameters to levels of the timbral properties. The user could specify a desired timbral character in terms of brightness, warmth etc. and the system evolved parameters for a synthesis algorithm whose sonic output matched this timbral character.

In contrast with the engineering, problem solving flavour of the above, Magnus used a GA as a conceptual part of the compositional framework, evolving raw time domain audio signals directly. [12], such that the evolutionary process became an explicit part of the aesthetic of the work as opposed to being a tool used in its creation.

Part of this work is technically similar to [8] and [11] in that it employs a GA to design a synthesis algorithm but it also infuses ideas from the area of interactive music systems [13]. In a pragmatic discussion of how to generate musically interesting results using such systems Bown and Lexer suggest allowing them to interact with an experienced musician who can guide the algorithm to interesting musical spaces, exploiting the skill of the musician and the exploratory power of an algorithm [14]. This work represents a fusion of these areas - a live algorithmic improviser which uses an unsupervised GA to design its sounds guided in real time by a live musician.

3 Technical Description of the System

The improviser can be split into two parts, the sound design system and the performance system. The sound design system evolves synthesis algorithms against audio input from the live musician which are used by the performance system to generate sound. See Figure 1 for an overview of the system.

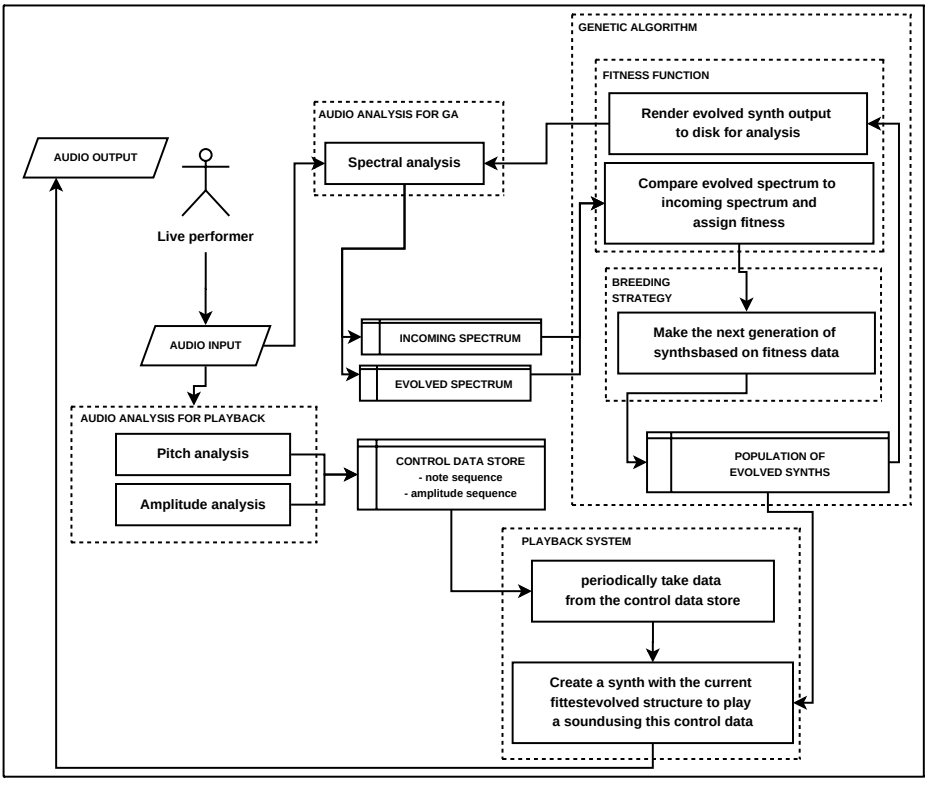


Fig. 1. An overview of the system

3.1 The Sound Design System

The sound design system has been built around the open source supercollider software [15] which provides two main components, sclang and scsynth. Sclang is an object orientated, interpreted scripting language specialised for music composition and sound synthesis functions. Scsynth is a sound synthesis server which responds to Open Sound Control (OSC) [16] messages sent via a network port by generating audio output in real time or it can process stored lists of OSC messages in an offline mode and render the output to disk as a sound file. Sclang can be used to define 'synthdefs' which are equivalent to the 'patches' found in other synthesisers or classes in an object orientated programming language. Synthdefs define which unit generators (from the many available in supercollider) are to be used in the patch and also arguments that can be used to change the state of the unit generators in the patch once it is running, e.g. change the pitch of an oscillator. Once a synthdef has been registered, by sending it to scsynth using OSC, it is possible to request that scsynth instantiates this synthdef so that the audio it generates can be heard. Since OSC is an open network protocol, like HTTP for example, a custom client can be written to control scsynth. This is what has been done here – the genetic algorithm has been written in Java, using scynth in its offline mode to render the output of the evolving sounds to disk for fitness analysis.

Synthesis Algorithms. The system offers two synthesis algorithms. The first implements time varying additive synthesis where the sound is made from many sine oscillators, each with its own 4 stage envelope. The envelope is applied to frequency or amplitude. The second is an FM system with modulator and carrier oscillators. Examples of these algorithms are shown in Figures 2 and 3. In the additive synth, each partial is generated from the output of an instance of the same synthdef. The arguments sent to this synthdef are shown in Table 1. The FM

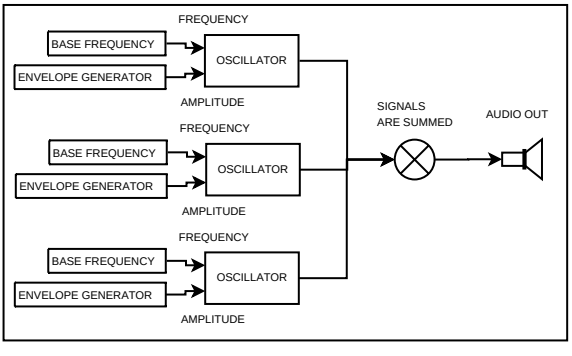


Fig. 2. An instance of the additive synthesis algorithm. Note that there is no redundancy in this synth graph – every unit generator affects the resulting audio signal.

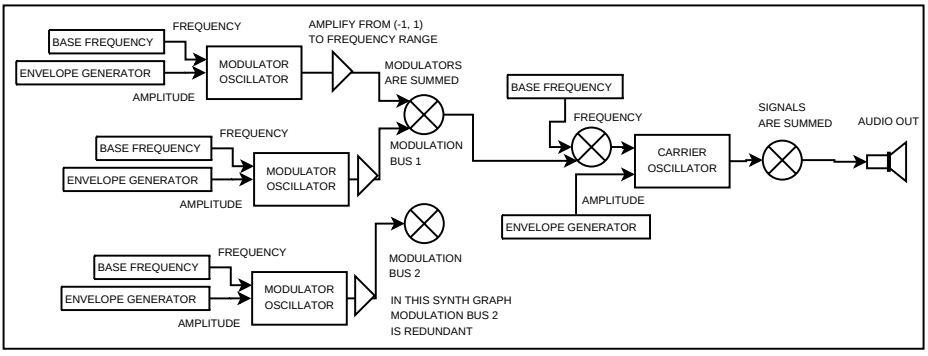


Fig. 3. An instance of the FM synthesis algorithm. Note that the genetic encoding permits redundancy in the synth graph where not every unit generator affects the resulting audio signal.

Table 1. Parameters for the synthdefs used to make the additively synthesized sounds. These parameters are derived from the numbers stored in the genome.

Name	Range	Purpose
masterAmp	0-1	Scales the amplitude of the oscillator
frequency	20-20000	The frequency of this oscillator in Hz
attackX, decayX, sustainX,releaseX	0-1	Peak times of the envelope
attackY, decayY,sustainY, releaseY	0-1	Peak values of the envelope
timeScale	0-10	Scales the length of the envelope
phase	0-1	The phase of the oscillator

synthesis algorithm utilises three types of oscillator, each of which is defined in a synthdef and the parameters for which are described in Table 2:

1. Carrier oscillators send their signal to the audio output. They have a base frequency which is modulated at audio rate by summing it with a signal read from a modulation bus.
2. Modulator oscillators send their signal to a modulation bus. Their frequency is fixed but an amplitude envelope fades them in and out.
3. Modulated modulator oscillators are like carrier oscillators except they write their output to a modulation bus instead of the audio output.

Genetic Encoding. For both synthesis algorithms, the genome is made up from an array of double precision floating point numbers with a fixed range of 1-10000. This array is parsed as fixed length segments or genes, each of which contains the parameters for a single supercollider 'synth', as defined in a corresponding 'synthdef'. An evolved sound is therefore made from the output of several supercollider synths (instantiated synthdefs), The values in the genome sequence are normalised into appropriate ranges for the unit generators they

Table 2. The parameters for the synthdefs used in the FM synthesized sounds

Name	Range	Purpose
synthDef id	1-6	Which synthdef to use (i.e. which type of oscillator)
masterAmp	0-1	Scales the amplitude of the oscillator
frequency	0-127	The oscillatorsbase frequency mapped exponentially to pitch inside the synthdef
frequency multiplier	1-4	An octave multiplier on the base frequency
frequency modulation bus	4-10	Frequency modulation bus
output bus	0-10	Output bus - could be a modulation bus or an audioout bus
envelope length	0-10	Envelope length in seconds, i.e. how long this oscillator is active

will be used to specify and configure. The genome length (number of genes) defines the number of synths that make up a sound and can increase or decrease, adding or removing complete sets of parameters from the genome. In the additive synthesis genome, this equates to adding or removing an oscillator.

Fitness Function. The fitness function follows this procedure:

1. Use an FFT to generate frequency bin power readings for the target sound, taking the number of contiguous frames as set on the control GUI. For example, if the snapshot count is set to 10, 10 contiguous, windowed FFTs are performed on the time series data, starting at the beginning of the target sound. Thus requesting a low number of snapshots will cause it to analyse only the start of the sound.
2. For each member of the population (individual), render its audio output to disk by decoding its genome to a command OSC file and parsing this with scsynth in offline mode, which will generate a PCM audio file.
3. Use an FFT to generate frequency bin power readings for this individual's audio file.
4. Compare the target frequency power readings frame by frame with those from this individual. For each frame, sum the squared differences between each frequency bins power reading then sum the differences for all the frames. Then reciprocate to get a measure of the closeness. This is the fitness awarded to this individual, which is thus the euclidean distance between the two spectral snapshotss.

Breeding Strategy. The breeding strategy is responsible for producing the next generation of sounds from the current generation. The program provides a few options here. The '2 seed, crossover, mutate and grow' option is the most complex. It runs the fitness function on the population, chooses the two fittest sounds then generates a population of new genomes by crossing over, mutating and growing the fittest genomes. The growing function allows a gene to be added

or taken away from the genome. The gene that is added can be derived by duplicating and mutating an existing gene or by generating a new random gene.

3.2 The Live Performance System

The improviser has a control system which is used to trigger and define sonic events. It is somewhat inspired after serialist composition, being based around ever changing sequences of numerical data that are used to configure the sound engine's output. The sequence data is initially generated in one of the following ways:

1. The incoming audio signal (e.g. from a live performer) is analysed in real time to produce readings of pitch and amplitude. These readings are saved to the 'control data store', where they represent a memory of the past behaviour of the live performer. The pitch data can be normalised to notes in the chromatic scale by a closest match search against several octaves of this scale, useful for melodic accompaniment.
2. Pitch data can be derived from a known scale and key, such that the control sequences cause the synthesis engine to play notes in that key.
3. Data can be generated at random.

This store of data sequences is constantly updated and this can happen in one of the following ways:

1. Data derived from one of the sources listed above is added to or used to replace a sequence.
2. An existing data sequence is transformed in some way. It can be inverted, it can be mutated or it can undergo crossover with another data sequence. The control data is manipulated on a per-parameter basis, e.g. pitch data is crossed over with pitch data. For example, the system may detect a series of notes played by the musician, play the notes back using evolved sounds then reverse the note sequence and play it back again.

The control system runs in the sclang environment. It continually instantiates fit members of the population on the scsynth server so they can be heard. It then passes control data from the stored data sequences to the running synths, which use it to update their current state. An example with the FM synthesis algorithm might run like this (see Figure 1):

1. The live musician plays a note. Its pitch is detected by the pitch analyser and this value is stored in a sequence with other detected pitches. Its amplitude is measured and stored in another sequence.
2. The control system instantiates the fittest sound from the current population so it can be heard and passes some frequency and amplitude data from the control sequences to all of its carrier oscillators which use it to update their base frequency and output multipliers respectively.
3. Meanwhile, the sound design system has captured a snapshot of the last note played by the live musician and is now silently evolving the population towards that timbre.

4. The live musician responds to the sound they hear.
5. After a certain number of events, the control data sequence might be transformed e.g. by scrambling the pitch sequence. Back to step 1.

4 Results

The two parts of the system warrant different assessments of their results. The timbre matching sound designer has been assessed on its ability to converge on fit sounds which match static test timbres. The live performance system introduces a moving target for the GA where the target timbre changes as new snapshots are taken of the live audio input – a very hard problem. The performance system's capabilities as an improviser can only be assessed subjectively.

Due to the recent addition of the FM synthesis algorithm, performance tests of the GA to date have been carried out using the additive synthesis algorithm with enveloped frequencies. Sounds were initially evolved using a spectral snapshot count of 40 with a single-seed, mutation-only breeding strategy where the next generation was formed from point mutated versions of the fittest genome from the current population. The target sound was a test sound generated at random with the same genome size settings. The system was found to increase fitness rapidly for the first 50-100 generations with a population size of 20. then to hit a local maximum, as expected from what is essentially a hill climber. Similar performance was observed with larger genomes. With increased spectral snapshots, the sounds were found to exploit the fitness function, converging on very short, quiet sounds. A length punishment was introduced where the ratio of the target sound to the evolved sound was used to reduce fitness and this prevented the short sound phenomenon. The optimal mutation method was found to be 10-20% of the genomic loci being mutated where the value is increased or decreased by a random amount between 0.1 and 10% of the total range (which was 1-10000). Higher mutation rates sent the population into somewhat random fluctuations in fitness, lower mutations reduced the convergence rate. Crossover was introduced to increase the parallelism of the search but even with larger genomes it was not found to increase the rate of convergence or the plateau fitness. This indicates that this genetic encoding scheme does not respond well to recombination and that for the additive synthesis model, using a GA offers little benefit over a hill climber with randomisation. A genome growing function was added as described in the breeding strategy section but performance tests with this feature have not yet been conclusive. It is hoped that genomic growth will allow the GA to match a complex target sound by starting from a small genome then increasing genomic size by adding useful new partials, which seems a simpler task than rearranging and mutating an initially complex genome.

Little numerical analysis has been done using the system in the live playback situation where the target sound is changed periodically. It is a possibility that the moving target will keep the fitness increase in the rapid hill climbing phase and that a higher mutation rate will be preferred to provide the variation the population will need to move towards new timbres as they become the next

target. The genomic growth function increases the variation in the population so may contribute to this effects as well. Regarding the system's improvisational skill, it does have a unique dynamic sound and the pitch and amplitude analysis give it a responsive feel. The FM synthesis algorithm provides a rich palette of varied sounds and can create atmospheric washes or rapid notes. The pitch analysis is based on the Pitch unit generator available within supercollider and is only designed to work with monophonic sources. However used in combination with the nearest match chromatic scale search mentioned earlier, it can pick out pitches successfully from a polyphonic source. Overall the improviser works well in a free improv type context.

5 Conclusion and Future Work

The fitness function described here transfers the time series into the frequency domain, which reveals more information about the timbral character of a sound than the time domain representation. The next step will be to take inspiration from the field of Music Information Retrieval (MIR), where highly efficient methods are being developed for the searching of large music databases for matches to a given sound or piece of music. One such system, [17] transforms time domain signals into sequences of variably sized 'audio lexemes' where an audio lexeme describes a short term spectral phenomenon such as a transition from a wide band spectrum to a low frequency spectrum. This approach uses the dynamics of the spectrum as opposed to simply its state at a given time. Using such an analysis may allow the improviser to respond to nuances in the timbre.

For real time timbre matching, a pre-evolved database of synthesis algorithms may be required. In this case, a database of synthesis algorithms would be evolved offline to match a variety of common musical timbres (e.g. timbres of instruments). Then for live performance, an MIR approach could be used to search for a synthesis algorithm from the database that is close to the incoming live signal then this could be evolved in real time against the incoming signal to refine the timbre match.

Acknowledgements

Thanks to Nick Collins and Chris Thornton.

References

1. Lewis, George E.: Too Many Notes: Computers, Complexity and Culture in Voyager Leonardo Music Journal (10) pp. 33-39 (2000)
2. William Hsu: Managing gesture and timbre for analysis and instrument control in an interactive environment NIME Proceedings, 376–379 (2006)
3. Tim Blackwell and Michael Young: Swarm Granulator Applications of Evolutionary Computing Volume 3005 (2004)

4. T. Takala, J. Hahn, L. Gritz, J. Geigel, and J. W. Lee: Using Physically-Based Models and Genetic Algorithms for Functional Composition of Sound Signals, Synchronized to Animated Motion International Computer Music Conference (ICMC) (1993)
5. Colin G. Johnson: Exploring sound-space with interactive genetic algorithms Leonardo, 36(1):51-54. (2003)
6. Sam Woolf and Matthew Yee-King: Virtual and Physical Interfaces for Collaborative Evolution of Sound Contemporary Music Review 22(3) (2003)
7. Holland: Adaptation in Natural and Artificial Systems University of Michigan Press (1975)
8. Thomas J. Mitchell, J. Charles, W. Sullivan: Frequency Modulation Tone Matching Using a Fuzzy Clustering Evolution Strategy AES 118th Convention, Barcelona, Spain, (2005)
9. Chowning, J. and D. Bristow: FM Theory and applications by musicians for musicians Yamaha Music Foundation Corp (1986)
10. A. Horner, J. Beauchamp, and L. Haken: Machine Tongues XVI: Genetic Algorithms and Their Application to FM, Matching Synthesis, Computer Music Journal, 17(4), 17-29. (1993)
11. Alex Gounaropoulos, Colin G. Johnson: Timbre interfaces using adjectives and adverbs, Proceedings of the 2006 International Conference on New Interfaces for Musical Expression (NIME06) (2006) 101-102
12. Cristyn Magnus: Evolving electroacoustic music: the application of genetic algorithms to time-domain waveforms Proceedings of the 2004 International Computer Music Conference (2004) 173-176
13. Robert Rowe: Interactive Music Systems. MIT Press, Cambs, MA, (1993)
14. O. Bown and S. Lexer: Continuous-Time Recurrent Neural Networks for Generative and Interactive Musical Performance EvoMusArt Workshop, EuroGP, Budapest (2006)
15. James McCartney: SuperCollider, a real time audio synthesis programming language http://www.audiosynth.com/ (last checked Nov. 2006)
16. Wright, Freed and Momeni: OpenSound Control: State of the Art 2003 Proceedings of the 2003 Conference on New Interfaces for Musical Expression (NIME-03), Montreal, Canada (2003) 153-159
17. Casey, M: Acoustic Lexemes for Organizing Internet Audio Contemporary Music Review special issue on Internet Music, A. Marsden and A. Hugil (Eds.). (2005)

Interactive GP with Tree Representation of Classical Music Pieces

Daichi Ando[1], Palle Dahlsted[2], Mats G. Nordahl[2], and Hitoshi Iba[1]

[1] The Univertity of Tokyo, Japan
dando@iba.k.u-tokyo.ac.jp
[2] Chalmers University of Technology, Sweden

Abstract. Research on the application of Interactive Evolutionary Computation(IEC) to the field of musical computation has been improved in recent years, marking an interesting parallel to the current trend of applying human characteristics or sensitivities to computer systems. However, past techniques developed for IEC-based composition have not necessarily proven very effective for professional use. This is due to the large difference between data representation used by IEC and authored classical music composition. To solve this difficulties, we purpose a new IEC approach to music composition based on classical music theory. In this paper, we describe an established system according to the above idea, and detail of making success of composition a piece.

1 Introduction

Compared with to use traditional stochastic composition technique, application of Interactive Evolutionary Computation (IEC) has some advantages.

In IEC methods, initialized population was generated randomly with methods given by user. Then the populations converge steadily passing through interaction between user and system. Finally the user can get results that are not needed to correct any more. In addition, IEC Systems apply genetic operations, such as crossover and mutation, basically randomly, hence the user can possibly discover unexpected promising results from the stochastic methods.

The most essential consideration when applying Evolutionary Computation to problem solving is the encoding that represents the problem as the gene. The efficiency of the creative evolution depends to a large degree on the representation utilized.

Also user interface and process of operation are important in Interactive EC. This is due to the problem that there are limitations to the population size and number of generation which user can deal with in IEC because of the user's burden. The previous researches tried to solve these problems of Interactive EC are described in [1,5,6,7].

For application to IEC for the sake of composition assistance, various gene representations and user interfaces have been tried. A general review of the application of EC, especially GA and GP, to composition can be found in [2]. About the user interface and operation, there are researches in order to implement user

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 577–584, 2007.
© Springer-Verlag Berlin Heidelberg 2007

operations to system as conversations between user and agent[1,4]. These experiment results have shown possibilities to reduce user's burden on evaluation which is a serious problem of Interactive EC. Simultaneously, there are various researches regarding gene representation of a musical phrase. Traditional composers often use tree topology to represent the result of analysis for their pieces. For this reason, tree representation of a musical phrase has advantages in the sense that it can easily be understood by the traditional composers in particular. Applying the presented IEC techniques in past researches, e.g. user refine or define genome directly and mask a part of genome manually, also shows the advantage of tree representation. In addition, tree representation that can represent musical repetition of typical classical pieces with recursive tree topology was proposed in [3], extending the tree representation of musical phrase.

However the representation of a musical phrase with tree topology has some problems. The problem is that tree topology easily becomes too complex to represent a comparatively long piece. In this case, the user - composer cannot understand the tree easily, in consequence, aplication of the IEC techniques that refine and define the genome manually becomes more difficult. Furthermore, dealing with complex and large trees degrades the performance of EC. For this reason, it was difficult to generate pieces with remarkable length with the systems that were present in past researches is difficult.

In order to solve these problems, we have constructed a new IEC system named CACIE(Computer Aided Composition using Interactive Evolution). CACIE is a system with the aim of aiding composers in the composition of traditional atonal pieces.

2 Construction of System

2.1 Gene Representation

Tree Representation of Musical Phrase. Tree representation was adopted as a gene representation to realize musical phrase and traditional compositional expression in a simplify the manner. Figure 1 shows the simple example of conversion of a musical phrase into a tree-topology. Each terminal node contains a note or a musical motive. In the first example of this figure, c, d, e and f are terminal nodes. A note contains four parameters: Note Number, Amplitude, Duration and Onset-Time. Zero amplitude represents a rest note. On the other hand, in the non-terminal node, functions that merge or concatenate the notes into larger musical structures, are represented as lists of notes. In the second example of the figure, S and U are non-terminal nodes. The S function connects two nodes being as leaf, lists of notes continuously. The U function merges two nodes that will be played simultaneously. The system provides any functions that realize traditional musical structure simply in addition to S and U. Table 1 presents a partial list of funcions that have been implemented. Besides, these functions are provided as libraries for user's programming.

Recursive Terminal Node. A special type of terminal node named Recursive Terminal Node, is a simple method of representing repetition of classical music

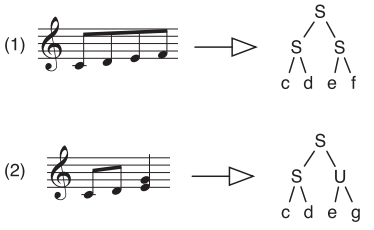


Fig. 1. Tree representation of musical phrase

pieces. A short genome with many notes of musical significance can be constructed using this kind of the node. Before the ontogeny phase when phenotype are created, recursive terminal nodes develop the tree recursively. As shown in Figure 2, the expansion of a recursive terminal node is displayed. Figure 3 gives a simple example of developing a musical phrase by the recursive terminal node.

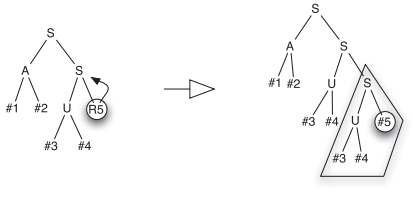


Fig. 2. Developmental process of a Recursive Terminal Node

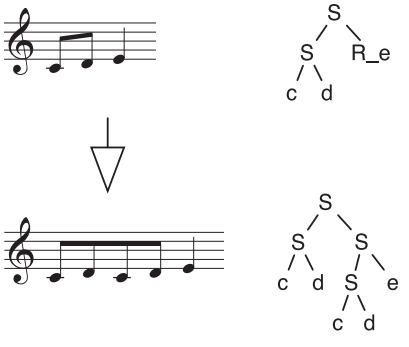


Fig. 3. Example of a Recursive Terminal Node

2.2 Genetic Operation

Additional special genetic operations we adopted are named Increase and Decrease. The increase mutation is a implementation of recursive terminal node, mentioned above, as mutation. Figure 4 shows example of an increase mutation. The reverse role of increase mutation is decrease mutation. The decrease mutation replaces a non-terminal node in a subtree with a terminal node.

2.3 User Interface and Composition Process

Multi-field User Interface. As mentioned before, as in figure 5, parent and offspring populations are displayed. Parent generation displayed in left side and offspring generation displayed in right side. The other window shown under the population window displays genome storage. We adopted the Multi-Field User

Table 1. List of part of functions

S	Connect two arguments continuously `(S a b) = (a b)`
U	Connect two arguments simultaneously `(U a b) = (ab)`
SR	Make a repetition of two arguments `(SR+5 a b) = (a b a b a)`

D	Apply rhythm pattern of 2nd argument to 1st argument
	`(D a(60,100,10) b(62,120,20))`
	`= a'(60,100,20)`

P	Apply pitch pattern of 2nd argument to 1st argument
	`(P a(60,100,10) b(62,120,20))`
	`= a'(62,100,10)`

A	Apply articulation pattern of 2nd argument to 1st argument
	`(A a(60,100,10) b(62,120,20)`
	`= a'(60,120,10)`

RV	Reverse ordering `(RV (a b)) = (b a)`

IV	Pitch Inversion
	`(IV a((60,100,10)(62,120,20))`
	`= a'((62,100,10)(60,120,20))`

TP	Pitch Transposing
	`(TP+5 a((60,100,10)(62,120,20))`
	`= a'((65,100,10)(67,120,20))`

MS	Return a sequence of arranged notes taken from two nodes alternately as follows: `(MS (a b c) (d e)) = (a d b c e)`
MU	Return a sequence of arranged notes taken from two nodes simultaneously as follows: `(MU (a b) (c d)) = ((U a c) (U b d))`
CAR	Return a sequence that containts the front X% of number of notes as follows: `(CAR50% (a b c d)) = (a b)`
CDR	Return a sequence that contains the rear X% of number of notes as follows: `(CDR50% (a b c d)) = (c d)`
ACML	Return an accumulated sequence `(ACML (a b c)) = (a a b a b c)`

FILP	Return a sequence that contains a repetition with pitch transposing until second node pitch as follows:
	`(FILP+2 (a(60,amp,dur) b(63,amp,dur))`
	`c(65,amp,dur))`
	`= (a(60,..) b(63,..)`
	`a'(62,..) b'(65,..)`
	`a''(64,..) b''(67,..) c(65,..))`

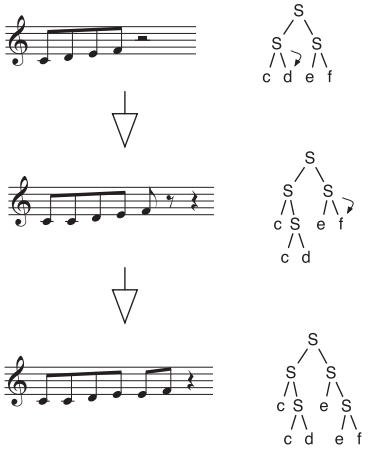


Fig. 4. Increase mutation

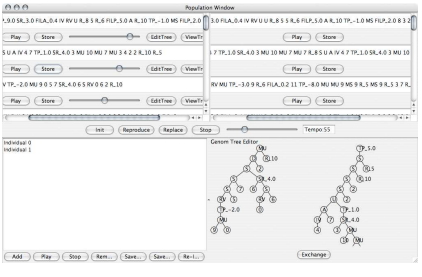


Fig. 5. Main window of the CACIE

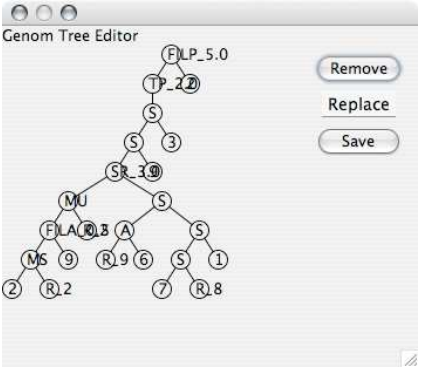


Fig. 6. Tree editor of the CACIE

Interface[7] in which offsprings are displayed in a separate window. Also, each individual presenting window have playback button and slider to give score. The user can compare and listen to the phenotypes of each individual of both parent and offspring populations before replacement. There is assumed to be a small population, thus the process of this comparison should not be very complex.

Manually Editing of Chromosome. A window shows in Figure 6 and a lower right space which shows two tree in the window of Figure 5 are tree editor space.

Sometime, composer want small changed which presented tree. However, it is difficult to get small changed tree which aimed by composer in composition with IEC. This is due to that genetic operatiors apply randomly.

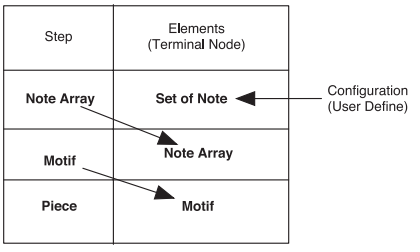


Fig. 7. Multiplex Structure: phenotypes of a step are used for terminal nodes of the next step

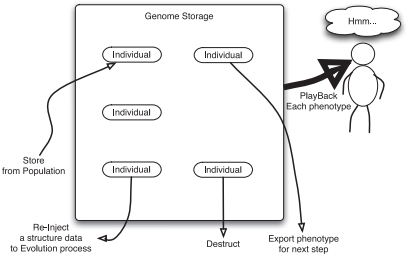


Fig. 8. Genome Storage: Storing the user's ideas

Thus, the authors implement mechanisms which realize that composer edit tree manually. Figure 6 is the window to edit one tree. With the window composer edit tree, such like replace a node or subtree with composer defined nodes or subtrees. Also in the lower-right space that shows in Figure 5, composer aplly subtree swap crossover manually.

Multiplex Structure of Composition Process. In traditional composing, short pieces such like the Invention or Lieder are typically started by deciding which notes to use for the piece. The notes are then ordered. In the next step, the composer combines the array of notes obtained in the first step and composes a "Motif" and motif variations. Lastly, the pieces are composed by transforming the Motive.

The CACIE includes this compositional "Multiplex Structure" method to the IEC process. In the CACIE, user composes a piece step by step. This multiplex structure is realized by the implementation, where each phenotype of the results run will be exported as MIDI event lists, then the user moves up to the next step including(by many) exported MIDI event lists as a terminal node. Figure 7 shows a diagram of the composition with the multiplex structure.

Genome Storage. A window shown in the lower left space of figure 5 is named Genome Storage. The genome storage is a temporary storage space by which strong intervention is realized for the user to evolution process as a part of multi-field user interface. The user can store individuals from the population in the genome storage anytime. Figure 8 shows an abstract of functions of Genome Stroage. Also, the genome storage has a function that import user define tree and export for multiplex structure.

3 Experiments and Discussions

3.1 Basic Developing

To confirm the effectiveness of the proposal representation, genetic operation and state of the evolving, a few functions were set, and the system was run

without operation using multi-field user interface and multiplex structure. In this experiment, for the terminal nodes, notes which contains the pitch C, E, G, A and C+(one octave upper) were used. The durations of all these notes were 8th. A functions, S and FILP were used. We confirmed the states of that genome were maturing musically with the application of increase mutation with generated melodies.

3.2 Evolving a Melody or Piece Step

Then, to verify the validity of multiplex structure, we have tried to generate a long piece with the system. In this case, we adopted a technique in which the user can refine the presented genome as well as store and re-injection. The acquired pieces are available from our Web Site as SMF(Standard MIDI File)[1]. We have confirmed that multiplex structure works effectively to reduce the user's burden when the user tries to generate a long piece at one time.

Lastly, we tried to case of the fully compose a piano miniature with the CA-CIE. The result contains large level musical structure, A-B-C-D-A'-B'-E. We can show this structure in the typical traditional classical musical pieces. Furthermore, each of the melodies to consists of the repetition of notes or small array of notes. Full musical score and performed SMF by professional pianists available from our Web Site[2].

3.3 Evaluation of the System

We asked any classical composers to evaluate the system, then to fill out a questionnaire in free style. We received several favorably points, also some unpopularity points.

The favorably points are as follows:

1. User can compose long piece by means of multiplex structure.
2. User can refer and revise musical structure directly.
3. Generating variations easily from only good offspring.
4. Tree representation of musical structure is easy to understand for classical composers.

On the other side, unpopular point was that programming user define function node is not constructed with GUI.

4 Conclusion

In this paper, we reported a research about our new system on applying Evolutionary Computing to computer aided composition system, in order that a traditional musical composer can use IEC system for their actual creation acively.

[1] http://www.iba.k.u-tokyo.ac.jp/ dando/public/projects/ecmusic/cacie/results.html
[2] http://www.iba.k.u-tokyo.ac.jp/ dando/public/works/rattfylla.html

Basic ideas of the presented system and gene representation are based on the traditional musical composition technique. As a result, we have succeeded to generate comparatively a long piece that include traditional musical expressions.

References

1. J. Biles. Genjam: A genetic algorithm for genrating jazz solos. In *Proceedings of 1994 International Computer Music Conference*, Arhus, 1994. ICMA.
2. A. R Burton and T. Vladimirova. Generation of musical sequences with genetic techniques. *Computer Music Journal*, Vol. 24, No. 4, pp. 59–73, 1999.
3. P. Dahlstedt and M. G. Nordahl. Augumented creativity: Evolution of musical score material, 2004.
4. B. L. Jacob. Composing with genetic algorithms. In *Proceedings of 1995 International Computer Music Conference*, Alberta, 1995. ICMA.
5. H. Takagi and K. Ohya. Discrete fitness values for improving the human interface in an interactive ga. In *Proceedings of IEEE 3rd International Conference on Evolutionary Computation (ICEC'96)*, pp. 109–112, Nagoya, 1996. IEEE.
6. N. Tokui and H. Iba. Music composition with interactive evolutionary computation. In *Proceedings of 3rd International Conference on Generative Art (GA2000)*, 2000.
7. T Unemi. A design of multi-field user interface for simulated breeding. In *Proceedings of 3rd Asian Fuzzy System Symposium: The Korea Fuzzy Logic and Intelligent Systems*, 1998.

Evolutionary Methods for Melodic Sequences Generation from Non-linear Dynamic Systems

Eleonora Bilotta[1], Pietro Pantano[2], Enrico Cupellini[3], and Costantino Rizzuti[1]

[1] Department of Linguistics, University of Calabria, Cubo 17b Via P. Bucci, Arcavacata di Rende (CS) 87036, Italy
bilotta@unical.it,
cosriz@yahoo.it
[2] Department of Mathematics, University of Calabria, Cubo 30b Via P. Bucci, Arcavacata di Rende (CS) 87036, Italy
piepa@unical.it
[3] Department of Mathematics, University of Torino, Via Carlo Alberto 10, Torino 10123, Italy
cupellini@dm.unito.it

Abstract. The work concerns using evolutionary methods to evolve melodic sequences, obtained through a music generative approach from Chua's circuit, a non-linear dynamic system, universal paradigm for studying chaos. The main idea was to investigate how to turn potential aesthetical musical forms, generated by chaotic attractors, in melodic patterns, according to the western musical tradition. A single attractor was chosen from the extended gallery of the Chua's dynamical systems. A specific codification scheme was used to map the attractor's space of phases into the musical pitch domain. A genetic algorithm was used to search throughout all possible solutions in the space of the attractor's parameters. Musical patterns were selected by a suitable fitness function. Experimental data show a progressive increase of the fitness values.

Keywords: Chua's Attractor, Genetic Algorithm, Generative Music.

1 Introduction

Mathematics and Music are strictly tied since the early foundation of Western culture. Since 80s, chaos and fractal geometry have strongly affected the development of a new field of musical research, with the use of non-linear dynamic systems for artistic purpose. Many musical researchers tried using non-linear dynamical systems as melodic pattern generators, able to "generate variation or paraphrase like alteration of specified groups of events" [1]. Bidlack [2] says that "chaos is of potential interest to composers who work whit computers, because it offers a mean of endowing computer-generated music with certain natural qualities not attainable by other means". According to Harley [3], non-linear functions and Music exhibit a comparable degree of self-similarity or autocorrelation.

Lately, modern researches on complexity use sounds and Music trying to develop new methods supporting traditional mathematical ways in understanding emerging

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 585–592, 2007.
© Springer-Verlag Berlin Heidelberg 2007

behaviours in complex and chaotic systems. In these years Rodet [4] used Chua's circuit for sound modeling, due to its non-periodical components; Bilotta et al. [5] used the same circuit for producing melodies.

In the following paragraphs we show how evolutionary methods could be used in order to manage the problems related to generative creation of music. Evolutionary music is a growing discipline that reached surprising successes. Khalifa, and Foster [6], for example, made a composition tool using Genetic Algorithms that generate and combine musical patterns with two fitness functions based on melodic intervals and tonal ratios.

The paper is organized as follows: Section 2 presents Chua's circuit and its attractors, describing also the codification process we used. Section 3 explains the Genetic Algorithm and fitness function formal aspects. Section 4 reports about the experimental results and musical analysis of these findings. Section 5 concludes the paper illustrating new directions of this work.

2 Chaos and Generative Music

2.1 Chua's Oscillator

Chua's oscillator is a non-linear circuit exhibiting a chaotic dynamical behaviour which provides a great family of strange attractors [7, 8]. Chua's circuit is a dynamical system with three grades of freedom and six control parameters: $\alpha, \beta, \gamma, a, b, k$; its dimensionless equations are:

$$\dot{x} = k\alpha[y - x - f(x)] \tag{1}$$

$$\dot{y} = k(x - y + z) \tag{2}$$

$$\dot{z} = k(-\beta y - \gamma z) \tag{3}$$

where:

$$f(x) = bx + \frac{1}{2}(a-b)\{|x+1| - |x-1|\} \tag{4}$$

In the initial part of this research we chose to focus our attention on one of the most famous Chua's attractor: the double scroll pattern (Figure 1). Table 1 shows the control parameter's values of this system.

2.2 Musical Codification

Generative music is based on two separate processes: an algorithm generating numerical sequences and a process for translating these sequences into melodic patterns. Since the coding process, also called *musification*, can be realized in a completely arbitrary manner, it is obvious that the quality of the musical rendering depends substantially on the choices which have been made. For this reasons, in defining a codification scheme, it's important to try realizing not only a mechanism that allows for a simple translation between numerical sequences and musical parameters, but also a codification system allowing for a meaningful transformations

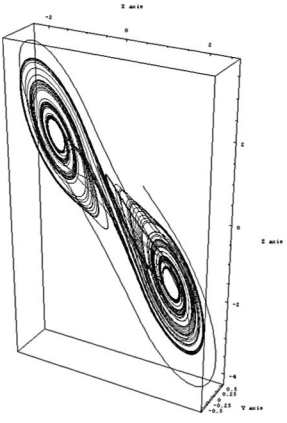


Fig. 1. Three-dimensional graphic of Chua's double scroll attractor in the space of phases

Table 1. Control parameters and initial values for the Chua's double scroll

α = 9,3515908493	β = 14,7903198054	γ = 0,0160739649
a = -1,1384111956	b = -0,7224511209	k = 1

preserving the main characteristics of the system's main features in order to exploit, from a musical point of view, the potentials dynamical systems have (fractal nature, different kind of behaviour and so on).

At the first stage we was concerned in defining a codification system to associate the solution *x(t), y(t), z(t)* of the system to a succession of musical notes S_k:

$$S_k = \{\text{note}_1, \text{note}_2, \text{note}_3, ..., \text{note}_k\}. \quad (5)$$

We have used a simplified codification scheme taking into account only the pitch parameter. Using a linear mapping (Figure 2) we translated into musical pitches, according to the MIDI protocol, the X-axis coordinate in the space of phases (see Figure 1). Moreover we chose to create a new MIDI note every time the waveform, related to the evolution of the X coordinate in the space of phases, reaches a

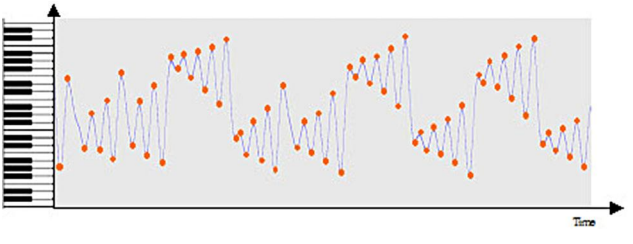


Fig. 2. Graphic schematisation of the musification process. The linear mapping transforms the numerical values of the attractor waveform in musical pitches.

maximum or a minimum point. Figure 2 shows a graphic schematisation of the used musical mapping; this kind of musical code can reveal, through the melodic pattern, some topological information about the dynamical system behaviour.

3 Evolutionary Process

In order to solve some of the problems related to the musification process, our system adopts a Genetic Algorithm (GA) search strategy to select melodic sequences obtained through a music generative approach. GAs are often cited as appropriate for exploring high dimensional parameter spaces as large regions of problems space can be covered quickly. Bilotta et al. [9] also presented a method based on a Genetic Algorithm to produce automatic music developing a fitness function based on consonance, which allows evaluating the "pleasantness" of a sequence of notes generated by an algorithm. In this work we chose to use the parameters ($\alpha, \beta, \gamma, a, b$) of the Chua's oscillator as a genotype and the melodic patterns, produced by mapping the X coordinate of the space of phases in a sequence of MIDI events, as phenotype. Table 2 shows the range in which every parameter can be varied:

Table 2. Range of parameters defining the searching space. These intervals were chosen to maintain the attractor's stability changing only one parameter a time. However changing more parameters at the same time the system's divergence from the double scrolling attractor (over flow) can occur.

8.80< α<9.73	14.10< β <15.08	-0.04< γ <0,10	-2.90<a<-1.09	-0.87<b<-0.62

In order to explain how the GA works, we'll give some formal definition.

Let $\kappa=\{\alpha, \beta, \gamma, a, b\}$ be a genotype related to a specific configuration of the attractor of the Chua's oscillator; we can define an initial population $P_\kappa^{(0)}$ of random generated genotypes in the searching space.

The GA scheme can be written as:

1. $\forall \kappa \in P_K^{(i)}$ calculates the melodic pattern related to the attractor;
2. $\forall \kappa \in P_K^{(i)}$ calculates the fitness value;
3. generates new genotype starting from the fittest genotype of the previous generation by applying crossing over, mutation and random generation;
4. builds $P_K^{(i+1)}$ and make it the current generation;
5. goes to 1.

This cycle is iterated until a fixed number of generations have been evolved.

3.1 Fitness Functions

The aim of the fitness function is to realize an automatic selection of the melodic patterns generated by the Chua's oscillator.

Let us suppose that a generative process has produced a sequence of n notes which we can group in a set X:

$$X=\{x_1,x_2,\ldots\ldots,x_n\} \tag{6}$$

If we assign to the lower C, belonging to the same octave the notes in X the label index 1, and the following $C^{\#}$ the label index 2 and so on until the note B, then we can define a set J of n indexes in the range (1-12) belonging to the notes in X.

We can define a fitness function which calculates the fitness value for the set of notes X as sum of all scores related to each melodic interval in the pattern:

$$f_{(J)} = \sum_{i=1}^{n} c_{J_i J_{i+1}} \tag{7}$$

Let us associate to each coefficient of the matrix shown in table 3 a constant value, the $c_{J_i J_k}$ coefficient represents the consonance score of the melodic interval x_i-x_k this value has been chosen according with aesthetic consideration about the melodic patterns we would to obtain from the evolutionary process. Let us call the obtained matrix C.

Coefficients of the C matrix must be chosen very carefully and some of them have to be negative; it becomes very clear that by choosing suitable values for the coefficients of the score matrix, consonant and dissonant melodic intervals, unison, chromatic or diatonic intervals can be discouraged or encouraged.

We used three different score matrices for separated evolutionary runs.
The first score matrix was based on consonance of successive tones:
Intervals of 3th, 6th, 4th , 5th , in the C major key, received a valuation of 2000.
Intervals of 2 th, 7 th, 8 th or unison in the C major key received a valuation of –100.
Intervals containing one note out of key ($C^{\#}$,$D^{\#}$,…) received a valuation of –100.
Intervals containing two notes out of key received a valuation of –1000.

Table 3. The first score matrix C

	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
C	-100	-100	-100	-100	2000	2000	-100	2000	-100	2000	-100	-100
C#	-100	-1000	-100	-1000	-100	-100	-1000	-100	-1000	-100	-1000	-100
D	-100	-100	-100	-100	-100	2000	-100	2000	-100	2000	-100	2000
D#	-100	-1000	-100	-1000	-100	-100	-1000	-100	-1000	-100	-1000	-100
E	2000	-100	-100	-100	-100	-100	-100	2000	-100	2000	-100	2000
F	2000	-100	2000	-100	-100	-100	-100	-100	-100	2000	-100	2000
F#	-100	-1000	-100	-1000	-100	-100	-1000	-100	-1000	-100	-1000	-100
G	2000	-100	2000	-100	2000	-100	-100	-100	-100	-100	-100	2000
G#	-100	-1000	-100	-1000	-100	-100	-1000	-100	-1000	-100	-1000	-100
A	2000	-100	2000	-100	2000	2000	-100	-100	-100	-100	-100	-100
A#	-100	-1000	-100	-1000	-100	-100	-1000	-100	-1000	-100	-1000	-100
B	-100	-100	2000	-100	2000	2000	-100	2000	-100	-100	-100	-100

The second score matrix is also based on consonance, but it's more selective because we have chosen score to widely discourage notes out of the chosen key.

Intervals of 3th and 6th, perfect 4th , 5th , in the C major key received a positive valuation of 20.

Intervals of 2th,7th,8th or unison in the C major key received negative valuation of -1. Every interval containing at least one note outside C major key received a negative valuation of -20.

Third score matrix only valuated if the notes belonged to the C major key, with a consequent positive or negative value (±2).

4 Results

All the experiments have been set up with a population of 40 elements, each one evolved for 100 time steps. From the selection process the 10 fittest melodies form the élite. To obtain the next generation we use:

- ten elements of the previous élite group;
- twenty elements obtained by crossing-over the elements of the élite with a random mutation of one gene;
- ten elements from a new random creation in the parameters space.

We have conducted experiments using the three score matrices above mentioned. The experimental results show that the genetic algorithms allows obtaining a best fitness values constantly increasing. In Figure 3 the graphics sketch the results of one run for each score matrix; graphs show the best and the mean normalized values of the fitness function.

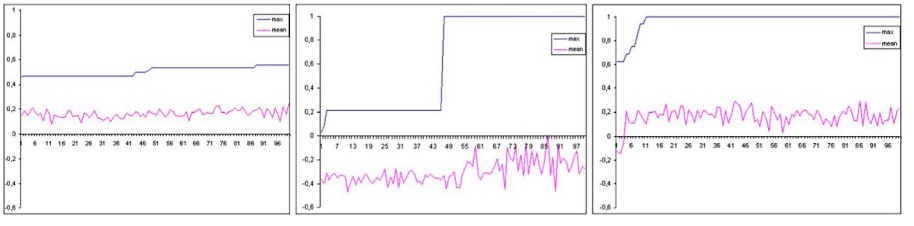


Fig. 3. Left: Results of the evolutionary run using the first score matrix. Center: Results of the evolutionary run using the second score matrix. Right: Results of the evolutionary run using the third score matrix.

Attractors of the first evolutionary process didn't reach the best values instead of others, probably because they were not properly selected. In fact, while first score matrix didn't evaluate negatively occasional notes out of the selected key, alternatively it gave a strong penalty to the passages between two pitches out of key.

4.1 Musical Analysis

Musical analysis for the first score matrix reveals the presence of short melodic patterns with a deviating contour (Figure 4). There are dissonant intervals even at early fragments of the melodies, that carry the listener out of tonality. Apparently the musical patterns seem to present a repetition of some parts of the entire structure, which manifests the fractal nature of the generating system.

The second score matrix caused longer melodic patterns (Figure 5). The orbit spends more time on a single attractor's lobe and then it scrolls to the other lobe. At least in the early sections of melodies there are consonant passages, and the melodic deviations can be heard like implicit harmonic progressions.

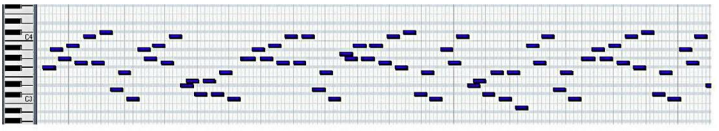


Fig. 4. Piano roll representation of the melodic pattern obtained by an evolutionary experiment using the first score matrix

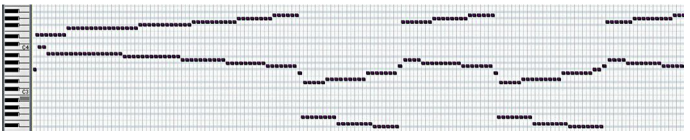


Fig. 5. Piano roll representation of the melodic pattern obtained by an evolutionary experiment using the second score matrix

Third score matrix didn't evaluate melodic intervals, it only evaluated if the notes belonged to the C major key (Figure 6). The attractors which produce these musical patterns were often narrow and their melodies were made of the same repeated pitches. Changes in melody pitches follow the scrolling movement of the orbit around a limit cycle. The musical effect is poorer but every pitch is in the C major key, so with different mapping range it could be possible to obtain different melodic jumps in the same key.

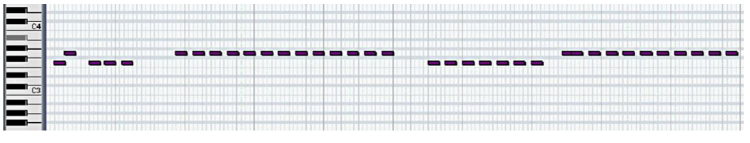


Fig. 6. Piano roll representation of the melodic pattern obtained by an evolutionary experiment using the third score matrix

5 Discussion and Further Progress

In this paper we presented a preliminary study about how to use Chua's circuit as a melodic generator of tonal sequences and Genetic Algorithms for searching out possible musical proportions in one specified attractor. The chosen musical codification, with a fitness function based on tonal progression have produced some interesting results. Two of the three selected score matrices led to melodies whose pitches were not every included in C major key. These findings suggest that not only the fitness function is important to make tonal progressions, using a pre-existing dynamical system, but it must be considered also the mapping range and maybe other musical codifications. In future works, we will try to operate a co-evolution of attractor's parameters and mapping extension. We are also interested in comparing melodic sequences from different coordinate in Chua's attractors space of phases, like Y and Z axes. Future developments of this work will use the evolutionary methods in order to evolve rhythmic and dynamic patterns selection.

References

1. Pressing, J., Nonlinear Maps as Generators of Musical Design. Computer Music Journal, Vol. 12, No 2 Summer 1988, MIT, (1988), 35-45.
2. Bidlack, R. Chaotic systems as Simple (but Complex) Compositional Algorithms. Computer Music Journal, Vol. 16, No 3, MIT, (1992), 33-47.
3. Harley, J., Generative Processes in Algorithmic Composition: Chaos and Music. Leonardo, Vol 28, No 3, (1995), 221-224.
4. Rodet, X., Vergez, C., Nonlinear Dynamics in Physical Models: Simple Feedback-Loop Systems and Properties. Computer Music Journal, 23:3, MIT, (1999), 18-34.
5. Bilotta, E., Pantano, P., Gervasi S., Readings Complexity in CHUA's Oscillator Through Music. Part I: A New Way Of Understanding Chaos. International Journal of Biforcation and Chaos, Vol.15, No. 2, (2005), 253-382.
6. Khalifa, Y., Foster, R., A two-state autonomous evolutionary music composer. EvoWorkshops (2006), 717-721.
7. Chua, L.O., Wu, C. W., Huang, A. & Zhong, G. Q., A universal circuit for studying and generating chaos. I. Routes to chaos. IEEE Trans. Circuits. Syst.-I: Fundam. Th. Appl. 40, (1993a), 732-744.
8. Chua, L.O., Wu, C. W., Huang, A. & Zhong, G. Q., A universal circuit for studying and generating chaos. II. Strange attractors. IEEE Trans. Circuits. Syst.-I: Fundam. Th. Appl. 40, (1993b), 745-761.
9. Bilotta, E., Pantano, P., Talarico, V., Evolutionary Music and Fitness Function, in A.M. Anile, V. Capasso, A. Greco editors, Mathmatics in Industry 1, Progress in Industrial Mathematics at ECMI, Springer, (2000), 127-139.

Music Composition Using Harmony Search Algorithm

Zong Woo Geem[1] and Jeong-Yoon Choi[2]

[1] Johns Hopkins University, Environmental Planning and Management Program,
729 Fallsgrove Drive #6133, Rockville, Maryland 20850, USA
geem@jhu.edu
[2] Washington Conservatory of Music,
3920 Alton Place NW, Washington, DC 20016, USA
jycpiano@yahoo.com

Abstract. Music pieces have been composed using a behavior-inspired evolutionary algorithm, harmony search (HS). The HS algorithm mimics behaviors of music players in an improvisation process, where each player produces a pitch based on one of three operations (random selection, memory consideration, and pitch adjustment) in order to find a better state of harmony which can be translated into a solution vector in the optimization process. When HS was applied to the organum (an early form of polyphonic music) composition, it could successfully compose harmony lines based on original Gregorian chant lines.

1 Introduction

Up to now, various nature-inspired or behavior-inspired algorithms have been applied to diverse fields. Genetic algorithm (GA), one of the popular phenomenon-inspired algorithms, has also been applied to music composition.

Horner and Goldberg [1] applied a GA model to bridge-music composition in the minimalist style. The fitness function in their model was the degree of pattern match and duration. Ralley [2] proposed another GA model to develop music melody. However, the melody developed by GA could not be evaluated because there was no appropriate fitness function. Biles [3] developed an interactive GA model, GenJam, to play jazz solos. The GenJam has been applied to many jazz tunes. Currently novel research between evolutionary algorithms and music compositions has been dealt with in several workshops [4-6].

In this study, another evolutionary algorithm (harmony search or HS), inspired by music improvisation, is applied to music composition in the medieval style, where a harmony line (vox organalis) is composed to accompany a given Gregorian chant melody (vox principalis). The proposed HS algorithm was created by analogy to the music improvisation process, in which musicians improvise the pitches of their instruments to obtain better harmony [7], and has been successfully applied to various real-world applications, such as truss structure design, water network design, traffic routing, and hydrologic parameter calibration [8-11]. The HS was superior to the GA in most cases because it overcame the drawback of the building block theory of GA [12]. This study

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 593–600, 2007.
© Springer-Verlag Berlin Heidelberg 2007

applies the HS to music composition problem which is to be formulated as an optimization problem with an objective function of medieval aesthetic, and the constraints of composition rules.

2 Harmony Search Algorithm

Music improvisation seeks to produce an ideal state as determined by aesthetic estimation. Similarly, algorithmic optimization seeks to produce an ideal state as determined by objective function evaluation. In the case of improvisation, the aesthetic estimation results from a set (harmony) of pitches produced by the music instruments involved, while the objective function evaluation is performed by a set (vector) of values in all the decision variables. Also, just as a harmony can be improved with each practice, likewise, the solution vector can be improved with each iteration (actually, human musician practice spans many songs, while optimization iteration spans only one song).

Figure 1 shows the analogy between music improvisation and optimization. Each musician (saxophonist, double bassist, and guitarist) is matched with each decision variable (x_1, x_2, and x_3). In addition, the range of each music instrument (saxophone = {Do, Re, Mi}; double bass = {Mi, Fa, Sol}; and guitar = {Sol, La, Si}) is matched with the range of each variable value (x_1 = {1, 2, 3}; x_2 = {3, 4, 5}; and x_3 = {5, 6, 7}), where the unit of the variables is meter if the variables stand for the pipe diameters in a water supply network.

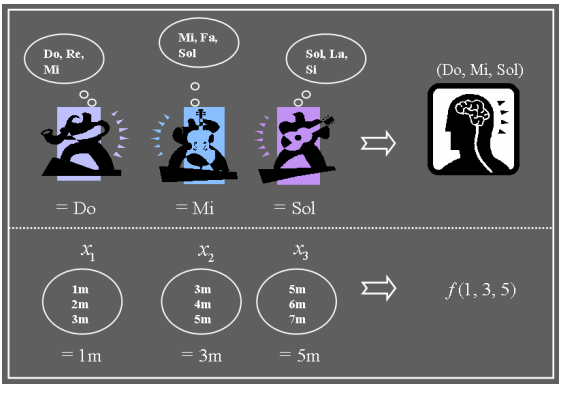


Fig. 1. Analogy between Music Improvisation and Optimization

Therefore, if the saxophonist plays the note Do, the double bassist plays Mi, and the guitarist plays Sol, their notes make a new harmony (Do, Mi, Sol). If this new harmony is "good" (aesthetically pleasing), the harmony is kept in the musician's memory. Likewise, the new solution vector (1m, 3m, 5m) generated in the optimization process is kept in computer memory if it is "good" in terms of objective function

value. Just as the harmony quality is enhanced practice after practice, the solution quality is enhanced iteration by iteration. The following is an explanation of the procedure:

2.1 Problem Formulation

First, the optimization problem is specified as follows:

$$\text{Optimize } f(\boldsymbol{x}) \tag{1}$$

$$\text{Subject to } x_i \in \mathbf{X}_i,\ i = 1,2,...,N\ . \tag{2}$$

where $f(\cdot)$ is a fitness function that evaluates the fitness of improvised harmony (most evolutionary music systems employ the human user as the fitness function); $\boldsymbol{x}$ is the set of each musical instrument (decision variable) x_i; $\mathbf{X}_i$ is the set of candidate pitches for each musical instrument, that is, $\mathbf{X}_i = \{x_i(1), x_i(2),..., x_i(K)\}$ where $x_i(1) < x_i(2) < ... < x_i(K)$; and, finally, N is the number of musical instruments.

2.2 Harmony Memory Initialization

Harmony Memory (HM) matrix, as shown in Equation 3, is filled with as many randomly generated solution vectors as HMS (harmony memory size).

$$\left[\begin{array}{cccc|c} x_1^1 & x_2^1 & \cdots & x_N^1 & f(\boldsymbol{x}^1) \\ x_1^2 & x_2^2 & \cdots & x_N^2 & f(\boldsymbol{x}^2) \\ \cdots & \cdots & \cdots & \cdots & \vdots \\ x_1^{HMS} & x_2^{HMS} & \cdots & x_N^{HMS} & f(\boldsymbol{x}^{HMS}) \end{array}\right] \tag{3}$$

2.3 New Harmony Improvisation

A new harmony, $\boldsymbol{x}' = (x_1', x_2', ..., x_N')$ is generated by following three rules: 1) random selection, 2) memory consideration, and 3) pitch adjustment.

Random Selection. Just as a musician produces any pitch within the instrument range (for example, {Do, Re, Mi, Fa, Sol, La, Si} in Figure 2), the value of the decision variable is randomly chosen out of the value range with a probability of freedom rate (FR).

$$x_i' \leftarrow x_i' \in \boldsymbol{X}_i = \{x_i(1), x_i(2),..., x_i(K)\} \quad \text{w.p.} \quad FR \tag{4}$$

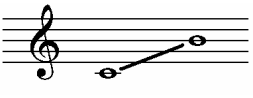


Fig. 2. Range of Musical Instrument

Memory Consideration. As a musician plays any pitch selected from the preferred pitches in his/her memory (for example, {Do, Mi, Do, Sol, Do} in Figure 3), the value of a decision variable x_i' is chosen from any pitches stored in HM ($\{ x_i^1, x_i^2, \ldots, x_i^{HMS} \}$) with a probability of HMCR (harmony memory considering rate, $0 \le \text{HMCR} \le 1$, HMCR = 1 – FR).

$$x_i' \leftarrow x_i' \in \{x_i^1, x_i^2, ..., x_i^{HMS}\} \quad \text{w.p.} \quad HMCR \tag{5}$$
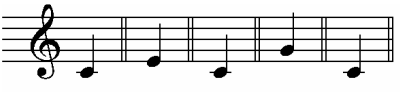


Fig. 3. Preferred Pitches Stored in Harmony Memory

Pitch Adjustment. Once one pitch is selected based upon memory consideration, a musician can further adjust the pitch to neighboring pitches (for example, the note Sol can be adjusted to Fa or La) with a probability of HMCR × PAR ($0 \le \text{PAR} \le 1$) while the probability of retaining the original pitch is HMCR × (1-PAR).

$$x_i' \leftarrow \begin{cases} x_i(k \pm 1) & \text{w.p.} \quad HMCR \times PAR \\ x_i(k) & \text{w.p.} \quad HMCR \times (1 - PAR) \end{cases} \tag{6}$$

Violated Harmony Consideration. Once the new harmony $\boldsymbol{x}' = (x_1', x_2', ..., x_N')$ is obtained using the above three rules, it must then be determined to confirm to harmony rules (= problem constraints).

If the new harmony violates the constraints, it may be used, but with a penalty. For example, the rule-violating harmony of parallel fifths was nonetheless used in musical works by accomplished composers such as Bach, Beethoven, and others. It should be noted that parallel fifths were actually abundant in the context of organum, which is the example used in this study. Parallel fifths are simply an example of the application of the general model.

2.4 Harmony Memory Update

If the new harmony vector, $\boldsymbol{x}' = (x_1', x_2', ..., x_N')$ is better than the worst harmony in the HM with respect to the fitness function, the new harmony is included in the HM and the existing worst harmony is excluded from the HM. If the stopping criterion (maximum number of improvisations) is reached, computation is terminated. Otherwise, another new harmony is improvised again.

3 Medieval Music (Organum) Composition

Gregorian chant is the monophonic unaccompanied sacred song of the Roman Catholic Church in the middle ages, and organum is an early form of polyphonic music

which accompanies the Gregorian chant melody. The HS algorithm is applied to the composition of organum by generating a harmony line (vox organalis) to accompany the original Gregorian chant (vox principalis).

Figure 4 shows the most ancient organum "Rex caeli Domine" contained in the anonymous book "Musica Enchiriadis" [13]. The upper line in the figure is a Gregorian chant melody and the lower line is the harmony line that was originally composed by an unknown person during the medieval era. The organum has several simple composition rules [13]: 1) the original harmony line progresses in parallel since it starts on the same note; 2) for the parallel motion, the interval of perfect fourth is frequently used while those of perfect fifth and unison (or octave) are also preferred; 3) in order to distinguish the vox principalis (chant melody) from vox organalis (harmony), the former should always be located above the latter.
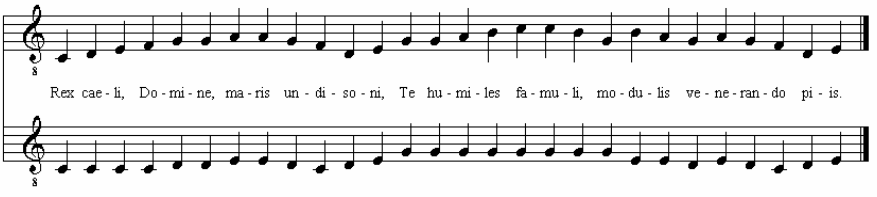


Fig. 4. Score of Organum "Rex Caeli Domine"

The composition techniques for the above-mentioned organum can be formulated as the following optimization problem:

$$\text{Minimize } \sum_{i=1}^{N} Rank(x_i) + \sum_{i=1}^{N} Penalty(x_i),\ i = 1, 2, \cdots, N \tag{7}$$

$$x_i \le m_i,\ i = 1, 2, \cdots, N \tag{8}$$

$$\left|x_i - x_{i-1}\right| \le \left|m_i - m_{i-1}\right|,\ i = 2, 3, \cdots, N \tag{9}$$

$$x_{Start} = m_{Start} \tag{10}$$

$$x_{End} = m_{End} \tag{11}$$

$$x_i \in \{\text{Do}, \text{Re}, \text{Mi}, \text{Fa}, \text{Sol}, \text{La}, \text{Si}, \text{Do}^+\} \tag{12}$$

where x_i is the i^{th} pitch in harmony line and m_i is the i^{th} pitch in original chant line. There are 28 pitches (number of decision variables, N = 28) required to create a composition in this study, which represents 8^{28} (= 1.93×10^{25}) combinatorial possibilities.

Equation 7 represents the fitness function, which is the summation of the rank term and the penalty term for each pitch in the harmony line (vox organalis). As the interval

of perfect fourth between vox principalis and vox organalis is most preferred, it receives the highest priority (rank = 1), as tabulated in Table 1. Table 1 shows the rankings for other intervals: a perfect fifth, unison (or octave), major or minor third (or sixth), and major or minor second (or seventh).

Table 1. Rank of Interval between Chant and Organum Pitches

Interval	Rank	Interval	Rank
Fourth	1	Fifth	2
Unison	3	Octave	3
Third	4	Sixth	4
Second	5	Seventh	5

For example, the rank for the interval of the first pitches in chant and organum lines in Figure 4 becomes three because they are unison (chant's first pitch m_1 = Do and organum's first pitch x_1 = Do), while the rank for the interval of the second pitches is five because they are major seconds (chant's second pitch m_2 = Re and organum's second pitch x_2 = Do). The lesser number (= higher rank) for the interval between chant and organum pitches is preferred because it has a smaller value in this minimization problem.

Equation 8 is the constraint that organum pitch be lower than or equal to chant pitch. If organum pitch is higher than chant pitch, a penalty value (= 5 in this study) is added to the fitness function.

Equation 9 is the constraint that the interval between two consecutive organum pitches be less than or equal to that in two consecutive chant pitches. If this constraint is violated, a penalty value (= 3 in this study) is added to the fitness function.

Equations 10 and 11 are boundary conditions that constrain starting and ending pitches in organum. Based on Gregorian chant's Latin verse, there are six notes that should be in unison (x_1 = Do, x_{11} = Re, x_{12} = Mi, x_{13} = Sol, x_{27} = Re, and x_{28} = Mi).

When applying the HS algorithm to organum composition with algorithm parameters (HMS, HMCR, and PAR are 10, 0.9, and 0.3, respectively, that are popular values in previous applications), the first improvised harmony does not appear pleasing as shown in Figure 5; it violates constraints many times (fitness value = 175).

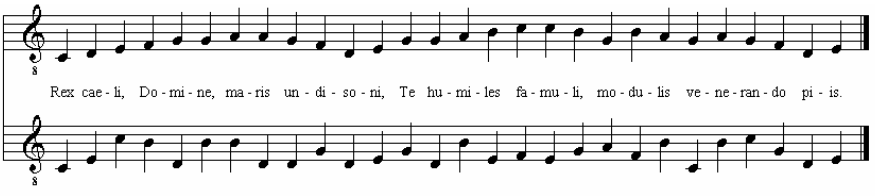


Fig. 5. Initial Organum (Fitness = 175)

After 3,000 improvisations, the HS algorithm found the organum with a fitness measurement of 42, as shown in Figure 6. It sounds satisfactory without any awkward pitches [14].
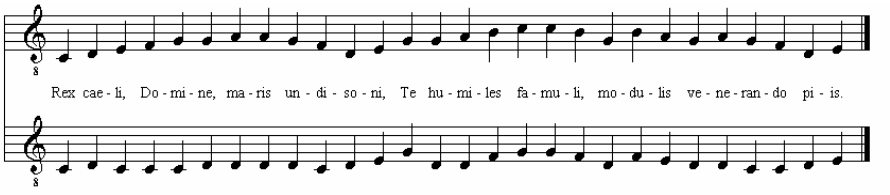


Fig. 6. Final Organum (Fitness = 42)

The HS was further applied to the organum composition based on the more complex (N = 50) Gregorian chant "Adoro Te Devote (Godhead here in hiding)." After 10,000 improvisations, the HS algorithm found the organum with a fitness measurement of 128, as shown in Figure 7.

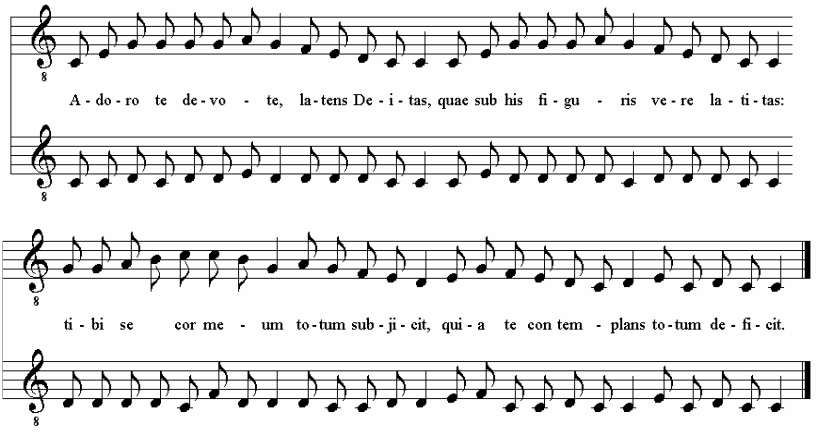


Fig. 7. Organum for Gregorian Chant "Adoro Te Devote"

4 Conclusions

A music-inspired algorithm, HS, has been applied to medieval music composition. The HS algorithm mimics the behaviors of musicians: random selection, memory consideration, pitch adjustment, and occasional violation of harmonic rules. These behaviors successfully created an organum composition in this study.

Applied to organum music composition, which was formulated as an optimization problem, HS composed satisfactory organum lines by generating up to 3,000 improvisations within one second on Intel Celeron 1.8GHz CPU. Total enumeration requires

8^{28} (= 1.93×10^{25}) function evaluations. Also, a more complex organum piece was successfully composed using the same process.

Future study of the HS algorithm should involve its application to more complex musical composition. Also, an efficient method to quantify the qualitative and subjective elements of aesthetic estimation in music composition should be developed.

References

1. Horner, A. and Goldberg, D. E.: Genetic Algorithms and Computer-Assisted Music Composition. Proceedings of the International Computer Music Conference. (1991) 437-441
2. Ralley, D.: Genetic Algorithms as a Tool for Melodic Development. Proceedings of the International Computer Music Conference. (1995) 501-502
3. Biles, J. A.: GenJam in Perspective: A Tentative Taxonomy for GA Music and Art Systems. Leonardo. 36(1) (2003) 43-45
4. Johnson, C. G. and Cardalda, J. J. R.: Genetic Algorithms in Visual Art and Music. Leonardo. 35(2) (2002) 175-184
5. Cagnoni, S., Cardalda, J., Corne, D., et al.: Lecture Notes in Computer Science (Vol. 2611) - Applications on Evolutionary Computing. Springer-Verlag, NY, USA (2003)
6. Rothlauf, F., Branke, J., Cagnoni, S., et al.: Lecture Notes in Computer Science (Vol. 3449) - Applications on Evolutionary Computing. Springer-Verlag, NY, USA (2005)
7. Geem, Z. W., Kim, J. H., and Loganathan, G. V.: A New Heuristic Optimization Algorithm: Harmony Search. Simulation. 76(2) (2001) 60-68
8. Geem, Z. W., Lee, K. S., and Tseng, C. -L.: Harmony Search for Structural Design. Proceedings of the Genetic and Evolutionary Computation Conference (GECCO). (2005) 651-652
9. Geem, Z. W.: Optimal Cost Design of Water Distribution Networks using Harmony Search. Engineering Optimization. 38(3) (2006) 259-280
10. Geem, Z. W., Tseng, C. -L., and Park, Y.: Harmony Search for Generalized Orienteering Problem: Best Touring in China. Lecture Notes in Computer Science. 3612 (2005) 741-750
11. Kim, J. H., Geem, Z. W., and Kim, E. S.: Parameter Estimation of the Nonlinear Muskingum Model using Harmony Search. Journal of the American Water Resources Association. 37(5) (2001) 1131-1138
12. Geem, Z. W.: Improved Harmony Search from Ensemble of Music Players. Lecture Notes in Artificial Intelligence. 4251 (2006) 86-93
13. Kim, M., Noh, Y., Park, M., Yi, S., Hur, Y.: Listening and Learning A History of Western Music – Volume I, SimSeolDang, Seoul, South Korea (1995)
14. http://www.hydroteq.com/

Curve, Draft, and Style: Three Steps to the Image

Olgierd Unold and Maciej Troc

The Institute of Computer Engineering, Control and Robotics
Wroclaw University of Technology
Wyb. Wyspianskiego 27, 50-370 Wroclaw, Poland
{Olgierd.Unold,Maciej.Troc}@pwr.wroc.pl

Abstract. This paper introduces a new evolutionary model of the art and design process. The whole process of interactive image creating was decomposed into three steps: evolving the Bezier curve, evolving the geometric transformations of the curve, and evolving the style of painting it. A model was implemented, and a series of promising experiments was performed.

Keywords: Evolutionary art, Interactive evolution, Bezier curve.

1 Introduction

We have observed a dynamic growth of evolutionary methods in all branches of computer science in the past few years. This development has already exceeded the boundaries of simple optimization problems and points to more ambitious goals, often associated with creative domains characterizing human activity. Since such a development has no theoretical limits, it has been noticed that evolutionary methods can also be used to create aesthetic values that are some forms of art themselves. In the ideal model of such a system, the evolutionary process should go autonomously, i.e., without a human help. Of course, encoding the appropriate set of aesthetic rules would be very difficult, that's why the human's help is needed to a certain degree during the process. Nowadays, the most popular form of human intervention is called interactive selection and it relies on placing the user in the role of the selector, who can choose the most aesthetically interesting images from individuals in the presented generation. According to an idea common for all types of evolutionary algorithms [6], the chosen best fitted images will create the next generation. This interesting technique can be practically used as an artistic and designing tool, because it leads to reaching some beautiful representations without manual work. Moreover, some complex (using mathematical formulas) masterpieces can be evolved in this way, which would be very difficult, or even impossible, with traditional methods.

If we treat evolutionary art as an artificial intelligence tool, it becomes obvious that the natural tendency in research tries to decrease the influence of man during the evolution. To make evolutionary art (evo-art) system really autonomous man can simply learn the way the beauty should be recognized. It can be done, for example, by "watching" the interactive evolution by a system and machine learning of appropriate rules. Some interesting research results on both aspects, i.e., interactive evolution and constructing an artificial artist, is described in the next section. After that we discuss

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 601–608, 2007.
© Springer-Verlag Berlin Heidelberg 2007

our concepts and introduce a model, in which interactive evolution has been used. In an experimental way, we try to prove the hypothesis that our tool can be used to reach some graphical aims planned by the user.

2 State of Art

The beginning of evolutionary art was examined by Richard Dawkins, who was the first to propose a concept of evolving of near-to-natural, visual, shapes with the help of man and his aesthetic sense. Though the idea described in [3] aimed mainly at showing the power of natural evolution, it quickly became a way of creating art. Based on this, numerous models have been proposed, probably the most popular one introduced by Sims [9]. He used Lisp expressions (evolved by genetic programming) to determine the relationship between the co-ordinates of a pixel and its color. This simple vision gained popularity and even today it is often researched. For example, in NEvAr [5] the model of evolution was, among other things, enriched with the possibility of conducting many independent experiments at the same time and exchanging data between any two of them. The use of Sims' structures is particularly interesting because of the fact that NEvAr's creators had experimented before with the model based on the Partioned Iterated Function System (PIFS) [4]. Thanks to this solution they could code any image, even a traditional photo, as a set of fractal transformations, modify such a structure in an evolutionary process, and store in KB.

At the same time, the new models of evolutionary art were introduced. Some of them are designed to be used in specific applications. Such a simple and interesting idea was proposed in [12]. The authors designed a tool which could be used to imitate the style of a well known Dutch geometric painter Pieter C. Mondriaan, based on straight lines, right angles, and simple colors.

Another significant system was Mutator – a tool for creating computer sculptures based on the procedural model [11]. Three-dimensional forms were generated there by an iterated transforming and placing basic (for example torus) or complex (for example "horn") solids. Some unexpected animal shaped representations (called organic artworks) were evolved this way.

Although the use of interactive evolution seems effective, the idea of building digital artists implementing some kinds of automatic evolution is not abandoned. The first attempt to apply such a system was taken by the Baluja's team and published in [1]. The structure of the genome was similar to that proposed by Sims, but the main goal was connected with testing neural network (NN) in a task of classifying attractive and ugly visualisations. The process of training NN was carried out with the help of the user. Baluja and co-workers tested five architectures of a neural network, but their results do not seem sufficient.

Another solution came with the placing process in a multiagent environment. The authors of [8] equipped every agent with the Kohonen's neural network, and the ability to communicate and new aesthetic values were achieved in consistence with the "law of novelty." The results of these experiments seem very interesting but the model in the examined shape (without human agents) is dedicated rather to modeling creativity then creating images itself.

It is worth to look at newer, successful projects in this area, for example [7] (co-authored by NEvAr creators). The authors have emphasized the module which judges artworks and classifies them into attractive and ugly ones. This subsystem, called Artificial Art Critic (AAC), was divided into two parts: "Feature Extractor" and "Evaluator". The first part pre-processes the artwork (musical or graphical) and extracts some features which can be important in the evaluation of the aesthetic value. The second one evaluates art using NN (like in works mentioned above, i.e., [1], [5], [8]). While "Evaluator" is based obligatory on a neural network, "Feature Extractor" may use any computational model, either a learning one or non-adaptive, that is programmed by a human.

It was not only NN that was used to construct an artificial aesthetic evaluator. In [10] a method based on a class of metrics called "normalized information distance" (making use of Kolmogorov Complexity) was applied. The researchers built knowledge base (KB) of images (with phenotypes and genotypes), which includes only highly aesthetically evaluated individuals selected by a human. Later, they compared each new image with those stored in KB. To carry out the comparison, two data sets, new image and the whole KB, were compressed into binary strings and the similarity between them was measured. Comparing either phenomes (images) or genomes (expressions similar to those proposed by Sims [9]) was empirically tested and for some parameters the algorithm achieved 75% accuracy in predicting aesthetic value.

Watching projects described before, we noticed the big popularity of the Sims' model [9]. It does create great possibilities, indeed. However, using this model to express shapes which were imagined by the user, seems rather difficult. Moreover, an automated aesthetic selection of such images is also a demanding goal. We think it is justified to use another structure generating less complex and less abstract visualizations and to take advantage of this improvement.

3 The Decomposed Model

Our model of visual representation and evolutionary process was based on the fact that even in the common sense of understanding visual art, it can be divided into two partially independent matters: the content of the artwork and the form in which it is presented. In our simplified way of understanding this division, we perceive the content as a set of visual objects (lines, shapes, etc.), which are graphical equivalents of the artist's ideas an the form is the way to visualize the primitives on the canvas, that is, it includes some information about colors, line's width, and additional effects, e.g., blending. The first component will be called "the draft", the second – "the style". We can use our incomplete, practical, understanding because it has been proven that human recognition of the content is based first of all on shapes [2].

Although the decomposition of the problem cannot be absolute, as both parts influence each other, we assume that evolving them separately can be in some cases effective. The user can focus only on one aspect of an image at a time. Moreover it is worth to mention that the interactive evolution can be used for quite small populations of individuals (dozen or so) and the decomposition leads to reduction of search space during a single process. Obviously, the decomposition of a static structure (phenotype

and genotype), as well as an evolutionary process pattern had to be done. For this reason our model is similar to Mutator [11], where the evolution of three-dimensional shapes is separated from the evolution of textures.

Finally we have decided to decompose the image into three parts (Fig. 1). Each of them is a separate element of visualisation (that's why we think about it as a part of phenotype) but also it is coded as a chromosome (separated piece of genotype). We will describe all three components briefly.

As a basic primitive the Bezier curve was used. This is because of the great possibilities this geometric formula offers in describing shapes, joined with simplicity of coding. It is easy to describe Bezier curve by determining the set of control points, which can be stored as a one-dimensional array including the co-ordinates. Additionally, two genes have been implemented. They define vertical and horizontal symmetry of a curve, because of the significant influence of these properties on the aesthetic value.

Only one Bezier curve placed on a canvas seems to be not sufficient. That's why we have decided to use some simple iteration model (similar to [11]). It is based on geometric transformations (like rotation, translation etc.) repeated a given number of times. Using that, we can cover the plain with the instances of one defined curve. Combining the definitions of the curve and iteration, we can get "the draft" mentioned in the previous section.

The last part of the image is a structure representing "the style". It has simple, linear form where particular parameters are placed one by one. Details of lines presentation such as: the colors, the way it's changing (gradient), the background color, the line's width and so on can be found there.

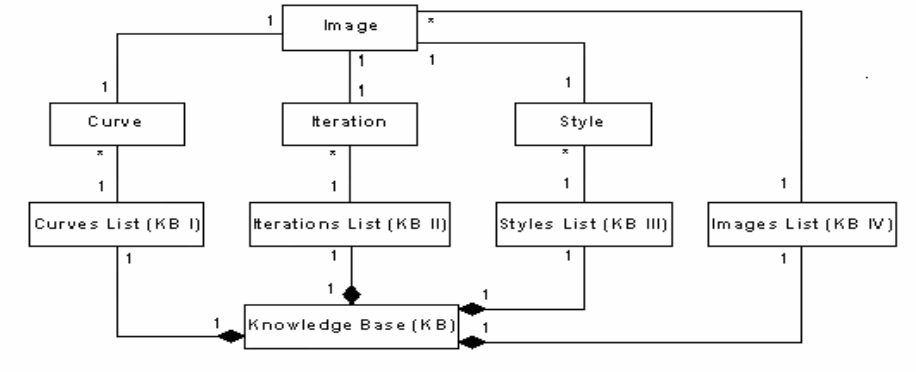


Fig. 1. The structure of the genotype and knowledge database (UML notation)

The structure of the genotype is shown in Fig. 1. To model it, the class diagram from the UML specification was used. The aggregation which characterizes the relationship between "the image" and all three basic elements can be easily observed. The object point of view has yet another justification. The genotype is divided not only in the structure but also in the set of operations. It means that the three parts possess a dedicated sets of evolutionary operators implemented specially for them.

Another interesting element is the knowledge base. In our model it exists in the distributed version according to the structure of genotype. We have introduced special sets, where we can store separated, interesting elements of images (KB I, KB II, KB III) and KB IV where whole genotypes are saved. Using that, we can build the images similar to those from the blocks. The user can easily use KB and exchange one of the three elements or build the parent of the first generation from the existing components. Because of this solution the modular structure of the genotype in our model was emphasized. In spite of the complex conceptual structure of the genotype, the physical realization is very simple. The whole image can be encoded as a linear structure with dynamic length.

As it was mentioned before, decomposition of the evolutionary process consists of three threads, which are dedicated to particular chromosomes. The particular elements of the image, also in our model, are strongly correlated and influence one another. That is why it is also necessary to create a possibility of evolving all three elements in one process. Finally, we decided that in the first thread only “the curve” is evolved, in the second - “the draft” and in the third - “the image”. The evolution of “the draft” has been implemented to modify “the iteration” but it can also be used to evolve only the curve or both elements in the same time (i.e., the whole “draft”). The third part of the evolutionary process was designed to shape “the style” but the whole image might also be evolved as its separated logic parts, i.e., “the curve”, “the iteration” and “the draft”. The scenario of the evolution depends on the user. Because the threads are performed on separated populations, the adequate migration mechanisms are necessary to be implemented. For example, we can take a chosen Bezier curve from the first thread and put it in place of the definition of the curve which is part of “drafts population” in the second thread. We can also choose one “image” and evolve “the draft” from it in a dedicated, separated thread.

When we know the general schema of the evolution, we can look at the construction of a particular thread. The experiments were performed for about a dozen or so individuals (10 - 16) in each population. As a basic type of selection a single choice was used (as in [3]). That is why we used an asexual replication and various kinds of mutation. Moreover we gave the user the possibility of changing the parameters of evolutionary process (as mutation probability) in every second (like in [11]).

4 The Experiments and Observations

To test the model and the application, a series of experiments was performed. First of all, the ability of shaping the Bezier curve using evolutionary algorithm was examined. We used the first thread of the evolutionary process, where all individuals in every generation represented a single curve.

The basic idea of evolutionary art models, based on interactive evolution, is to let the user lead the process according to his or her ideas. To do that, we tried to evolve some simple symbols starting from random generations. We aimed at achieving shapes such as: “fish” (Fig. 2), “heart” (Fig. 3) or “butterfly” (Fig. 4).

The results of the first experiments were promising. We noticed that we can easily lead the interactive evolution to reach some simple shapes and the efficiency of this activity was strictly connected with the experience of the human creator who was

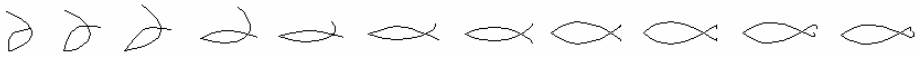


Fig. 2. Some chosen stages in the evolution of the fish shape



Fig. 3. The heart shape defined with seven (a), ten (b) and sixteen (c) control points. The first curve (random), one from the middle of the process, and the last one are presented.

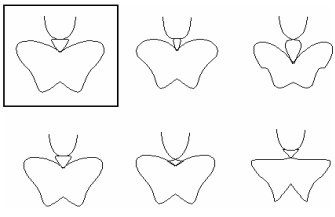
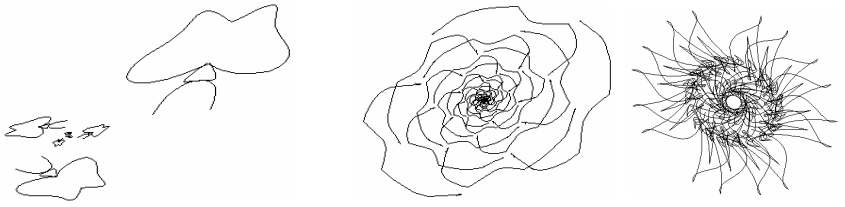


Fig. 4. The parent and some children chosen from one population during the generation of butterfly shape

working with the program. The evolutions of the fish shape took about 40 generations (selected steps in Fig. 2). Unfortunately, the efficiency decreased significantly as the number of control points increased. In Fig. 3 three heart shaped symbols are shown. We can reach it starting from random curves, but the larger the control points set, the more difficult leading the process the deterministic way. The evolution of a butterfly shape (Fig. 4) was possible due to the mechanism of modifying the ranges and probabilities of evolutionary operators. For example, when the user began the process, he or she can set high values to search the wide space of curves and to find the most promising shape (exploration phase). After that, they could reduce the limit parameter and try to improve the basic image (exploitation phase). In Fig. 4 the simple butterfly shape was already enriched (parent curve) and the user is trying some variations. The value of the probability operator is set at 12 % and the maximal range was increased to 56 (max. 100). Because only a small part of genes (control points co-ordinates) was changed in every descendant, the range was quite big and some details in the parent shape were significantly modified.

As a next step of experiments we tested the draft evolution. First of all, we followed our basic idea and tried to fit an appropriate iteration schema into the curve we had created before. We thought that if we describe a given object using the Bezier curve, we would put groups of its copies on the canvas in some interesting configuration. For example, we thought that we could create something like a “butterfly swarm”. Although we evolved an appropriate schema (Fig. 5a), we noticed that the proposed iteration model can describe not all of the interesting configurations.

The basic, linear, geometric transformations are not sufficient for this purpose, and more complicated model is needed. Nevertheless, we noticed that the evolving curve and iteration together gave interesting results. We could evolve some objects according to our intentions, but they would have to have a regular shape (round or elongated) and a properly described, by iterated geometric transformations, repeating property (Fig. 5b, Fig. 6).



a) b)

Fig. 5. Round visualizations: "Butterfly swarm" (a), "Rose" and "Sunflower" (b)

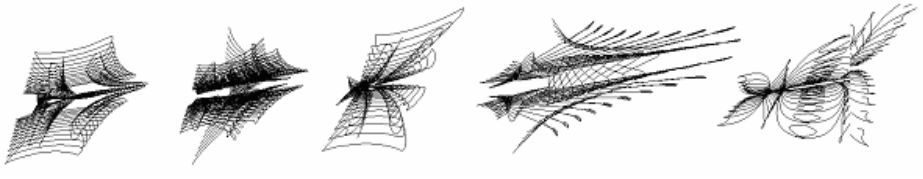


Fig. 6. Some „flying objects" evolved from a common ancestor

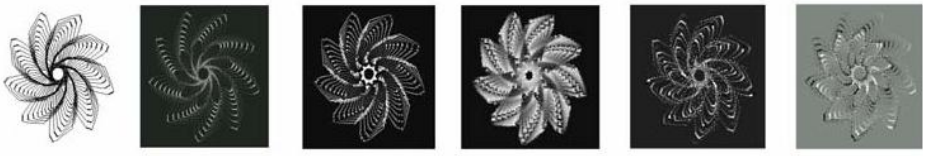


Fig. 7. The draft and various images based on it

In the last series of experiments, we examined the style and whole images evolution. In Fig. 7 six elements are shown: one draft and five images which were created by the evolution of style from this draft. We can observe the relation between those two components. The presented draft is not very interesting from the aesthetic point of view but by just adding the appropriate style definitions we can reach some nice visual effects. The ready, complete, images differ sometimes significantly, but the basic shape represented by the draft is visible in every image. Looking at them, we are convinced that separating the draft, which represents the content, and the style, which represents the form, may be a beneficial solution compatible with natural ways of looking at art. We can also save interesting drafts in knowledge base (KB III) and look for some appropriate styles for them.

5 Summary

We can distinguish three vital properties of the introduced model: (1) dividing both the structure and the process into differing parts, (2) expressing ideas with the help of lines, (3) letting the user influence the evolutionary process significantly.

For the sake of: the decomposition, the use of iterated transformations and the significance of the possibility of changing evolutionary parameters by the user, our model is similar to some other solutions, for example [11]. Nevertheless, many changes were introduced to reach our main goal, that is, letting the user generate shapes, which they planned before the start of the evolution. We extended the conception of the image decomposition into different parts, and we used the Bezier curve as a basic primitive. We suppose that this model could be appropriate to implement a learning module of automatic selection, too. The set of curves (control points coordinates) contains more substantial information then a bitmap and can be analized easier.

References

1. Baluja, S., Pomerlau, D., Todd, J.: Towards Automated Artificial Evolution for Computer-Generated Images. Connection Science 6 (2/3) (1994) 325-354
2. Barrow, J.D.: The Artful Universe. Oxford University Press, Oxford (1995)
3. Dawkins, R.: The Blind Watchmaker. W.W. Norton & Company Inc., New York (1987)
4. Machado, P., Cardoso, A.: Model Proposal for a Constructed Artist. In: Callaos, N. at all (eds): Proc. SCI'97/ISAS'97. Caracas, Vol II (1997) 521-528
5. Machado, P., Cardoso, A.: NEvAr - The Assessment of an Evolutionary Art Tool. In: Wiggins G (ed.) Proc. AISB'00, Birmingham (2000)
6. Michalewicz, Z.: Genetic Algorithms + Data Structure = Evolution Programs. Springer Verlag, Berlin Heidelberg New York (1996)
7. Romero, J., Machado, P., Santos, A., Cardoso, A.: On the Development of Critics in Evolutionary Computation Artists. LNCS 2611 (2003) 559-569
8. Saunders, R., Gero, J.: The Digital Clockwork Muse: A Computational Model of Aesthetic Evolution. In: Wiggins G (ed.) Proc. AISB'01, York UK (2001) 12-21
9. Sims, K.: Artificial Evolution for Computer Graphics. Computer Graphics 25(4) (1991) 319-328
10. Svangard, N., Nordin, P.: Automated Aesthetic Selection of Evolutionary Art by Distance Based Classification of Genomes and Phenomes Using the Universal Similarity Metric. LNCS 3005 (2004) 447-456
11. Todd, J., Latham, W.: Mutator, a Subjective Human Interface for Evolution of Computer Sculptures. IBM United Kingdom Scientific Centre Report 248 (1991)
12. Van Hermet, J.I., Eiben, A.E.: Mondriaan Art by Evolution. In: Postma, E., Gyssens, M. (eds.) Proc. BNAIC'99 (1999) 291-292

GISMO2: An Application for Agent-Based Composition

Yuta Uozumi

Cyber Sound Project, Media Design Program Graduate School of Media and Governance,
Keio University SFC, Japan
isana137@sfc.keio.ac.jp

Abstract. This paper presents an approach to new method for music composition with multi-agent system, which is a field of new research and offers vast possibilities for both scientific research and artistic expression. Proposed model is an ecological approach in which the agents coexist in a virtual world displaying predator/prey behavior. It realizes self-organization of sound structure and real-time performance with laptop PC. The demo movies, including all reference movies in section 3 are available on the Web (http://www.dubdb.com/gismo/evo/). Please refer to them for agents' behaviors and sounds.

1 Introduction

Gismo is an application to compose music with multi-agent system. The multi agent system is simulation technique for complex system [1]. It can self-organize[2] musical structure through interactions between agents.

"Swarm music", which was implemented by Blackwell[3] already exists as representative study of improvisation with multi-agent system. It focused on application of insects-swarm behavior to musical field. However such existing models are too complex for general user to intuitively catch up and operate.

Gismo focuses on vast possibilities of multi-agent system as universal interface for musical composition. User Interface should be simple, clear and intuitive. Proposed model was designed for these requirements, therefore composers can easily intervene interactions between agents in real time.

In the style of traditional Western music, composers had to design all musical factors based on its theory. However with gismo, the composers only have to design musical-components as agents and make them interact with each other. Then, composers operate the application according to result of those interactions in real-time. The new style combines composition with improvisation.

2 System Component

2.1 Development and Execution Environment

This application is developed with the C and Objective-C programming language. Development environment is Xcode 1.1 made by Apple Computer Inc [4].

This application requires OSX 10.4 above.

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 609–616, 2007.
© Springer-Verlag Berlin Heidelberg 2007

2.2 Model, Framework

In the field of the multi-agent system, various models have already been proposed. This application employs simple an ecosystem model optimized for Gismo.

The users should understand what is going on in interactions between agents immediately, and operate the application to react to them. That is reason why the adoption model should be simple. The model is the simplest model that connects each agent in relation of food chain. Events "chase" or "escape" occur in accordance with it.

Each agent has its own parameters such as field of vision(View), size(Size), movement speed(Mov). The algorithm of agent is as follows (Figure 1).

1. At first, agents walk around randomly in virtual world.
2. When an agent finds the others, it compares its size with the nearest one. If the other is smaller, it chases and eats the other. If the other is larger, it escapes from the other to survive.
3. When the agent attacks the others, it gets bigger (increase in SIZE parameter). On the contrary, when it is attacked by the others, it gets smaller (decrease in SIZE parameter). If the value of size decreases below the pre-defined value, the agent dies.

This is very simple, however, the essential factors exist there, such as conditional judgment, positive-feedback, negative feedback, and increase and decrease in options, etc. Since the behaviours of agents are visually apparent on this model, the users can understand what is happening in the virtual world.

2.3 Sound

In this application, users can set arbitrary sounds to be triggered by various events (Table 1). This is done by reading and playing of sound files. Users can utilize AIFF, WAV and SND format sound files.

When the state of agent changes, the sound set up on each event is played. Consequently, the interactions between agents generate rhythms and/or harmony. The parameters of agents influence repetitions, rhythms or harmonies.

For example, in the case where agent repeats “finding and chasing” and “missing”, it generates rhythm. If the MOV parameter(movement speed) of the agent is very high, it might generate the effect similar to fast BPM(beats per minute).

Table 1. Sound event Chart

Event Name	Timing
Calm	Losing sight of other agents
Escape	Escape from other agents
Chase	Chase other agents
DMG	Damaged
Death	Agent death

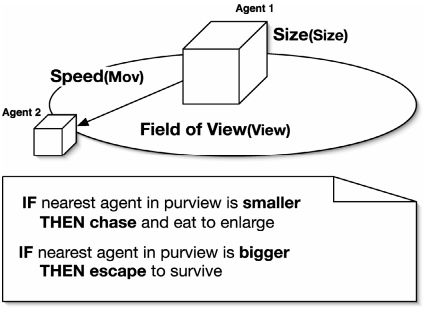


Fig. 1. Basic Model

2.4 GUI

The interface is implemented for framework of gismo (Section 2.2), which is designed for users in order to operate the application quickly and smoothly in response to agents' actions (Figure 2).

The contents are as follows.

WorldView: This area displays aspects of virtual space in gismo. Various behaviors of agents can be seen here.

PalletteInterface: This area is interface for defining new agent species. When the users click on "+" button, a new agent species is added and all parameters are set to their default values. The users can reconfigure these parameters here to design the new agent species. Also users can assign a sound for each event, such as escape, chase, damaged, death.

AgentPutButton: To put the agent designed within PalletteInterface into virtual world, the users need to push this button. They can add as many agents as they want.

AgentEditer: The agent put into the world can be controlled with AgentEditer, and it affects the agents immediately.

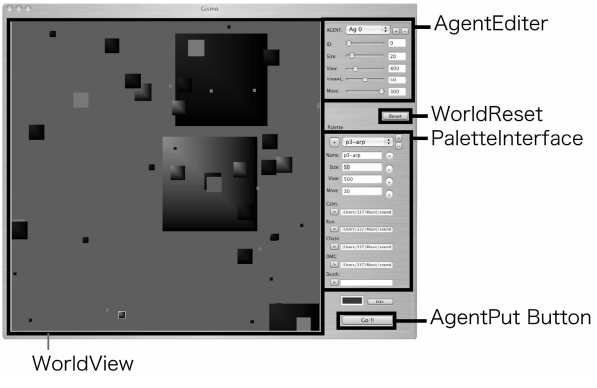


Fig. 2. Gismo Interface

3 Samples of Implementation

Let us focus on actual example with the application.

3.1 One to One: The Simplest Case

The interaction of between two agents is the simplest case in Gismo. In this case, two identical agents are put into the world. When each agent finds the other, both agents chase the other to eat(because their sizes are same), in another second, winner and loser are decided. Consequently, the loser escapes, and the winner chases. This relation is static in this case.

Another notable matter is "loop processing". Gismo processes all sides of virtual world as loop (Figure 3). Consequently, it generates pseudo metrical structure based on

loop structure built by agents that move endlessly in loop of the world. In other words, the escaping agent comes at the corner of the world, and then it moves to the counter side by the loop process. At this time, they lose sight of each other. Next, the chasing agent comes at the corner too, and then it moves to the counter side in the same way, and they find each other again. Then Gismo plays sound which was assigned to this event beforehand. If this process is repeated, it can generate looped sound.

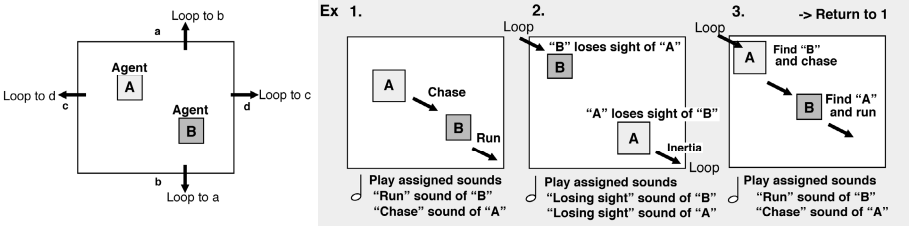


Fig. 3. Loop of Virtual World

3.2 Many-Body Case

Let us discuss many-body case. In this case, a number of agents are put into the world according to the following composition(Table. 2). We see from table 2 that smaller agents in size are larger in number and quicker in move, and that bigger agents in size are fewer and slower. However, the agent "Trialist" is an exception. It is an examinee agent that is especially set. Trialist's "size" parameter is same as "lower" agent, but the other parameters such as movement speed and field of vision are different, and they are much higher than those of the other agents. Thus, Trialist has advantages, and consequently, the agent can eat the other agents with its advantages and becomes larger (Figure 4-1). Before long, the agent can eat "medium" or "higher" agent which were bigger than it at the first(Figure 4- 2, 3).

Moreover, Trialist agent is set for playing white noise when it is chasing after the other agents. Therefore, as the agent grows and its option to eat the others increases, repetition of white noise increases. Also, by setting Trialist to play decaying sine wave when it escapes from larger, we can see the decrease of its threat as repetition of this sound decreases. Of course, we can set some other sound instead of white noise or

Table 2. Example of many-body composition

Agent Name	Size	Mov
Lower	10	50
Medium	30	30
Higher	60	10
Trialist	10	300

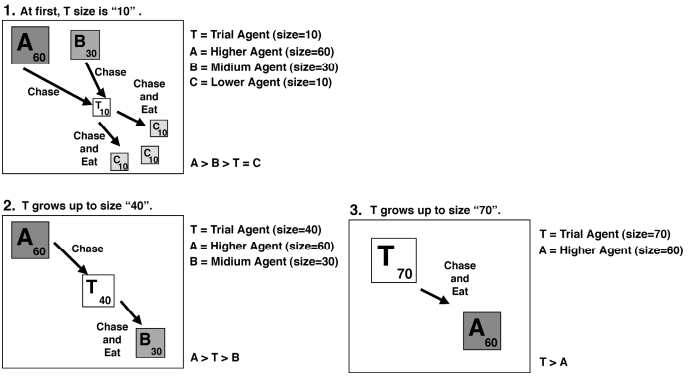


Fig. 4. Growing agent

sine wave. For example, it is possible to generate harmony by assigning C and G sound to each agent.

3.3 Beat Generation

Improvisation music like jazz uses fluctuation beats which, for example, ride-cymbal generates. Gismo can generate it with self-organization by agents.

In this case, beat agents are employed (See Table 3).

If twelve (or more) agents are put into the world, gismo can generate fluctuation beats. The features of this agent are "ultra-narrow field of vision(View parameter is super low)" and "Very fast movement speed(Mov parameter is super high)".

Beat generation process is as follows.

1. At first, beat agents cannot find each other because of their super-narrow field of vision, therefore, they walk around randomly.
2. Before long, one agent finds the other and then chases or escapes. According to this event, sound is played.
3. However, they lose sight of each other easily because they have too narrow view of field and move too quickly.

If escaping agent enters the field of view of another agent, this sequence of interactions occurs again.

Table 3. Beat generation

AgentName	Size	Mov
Ride	20	300

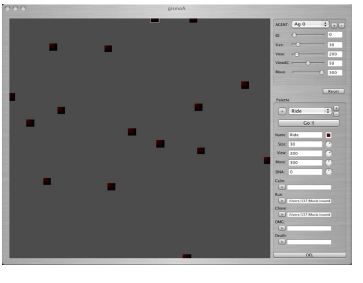


Fig. 5. Self-organization by Beat-agents

This simple process brews up a chain of interaction. As a result, it generates complicate fluctuation beats. Furthermore, agents self-organize, so that they keep the constant distance among them(Figure. 5).

3.4 Swarm Implementation

The Gismo can make agents swarm.

Interaction speed of swarm algorithm is so fast that it can realize granular synthesis[5]. Boids by Craig W. Reynolds is known for model of swarm or flock[6]. However, Gismo adopts simple algorithm like fly's behavior. This method has a defect that a swarm breaks up more easily than the method of Boids. Instead, it is compatible with the ecosystem model of Gismo and can process more rapidly. Swarm algorithm is as follows.

Firstly, the agent compares its species with the nearest other agent and checks whether it is same or different. Secondly, when they are the same species, the distance between them is compared with the fixed value of system. Thirdly, if the distance is greater than the value, they get close to each other, or vice versa (Then gismo plays sound in response to these event.). Therefore, those agents keep constant distance based on the fixed value. Consequently, agents generate swarm behavior like flies.

This algorithm is very simple, however, it generates complicate behavior with the interactions between agents. In this case, if we assign short sound files (30 - 120msec) to agents, we can get interesting timbres. However, granular synthesis is too heavy for the present version of Gismo to generate sound with it. It is a next task that is necessary to optimize the sound implementation.

4 Evolutionary Factors in the Application

4.1 Mating of Agents with Genetic Algorithm

Species types are set to each agent by user. If agent finds other that has same species type, they swarm(Section3.4). Then, these agents generate swarm behavior like flies. During this time, each agent increases each value of familiarity parameter. If the value increases and exceeds the pre-defined value without being interrupted by predators, the agents mate and divide(generate offspring) based on genetic algorithm(Holland 1992). The offspring inherit parent's features with crossover and mutation in the genetic algorithm. Accordingly the performance generates new evolution.

4.2 Network Communication with OSC

Gismo mounts network communication function. OSC protocol is adopted to the implementation, which is proposed by CNMAT Technologies, U.C. Berkley [7].

Two users can make musical session with the function. Utilized PCs are connected with each other by wireless or wired peer-to-peer communication. Each PC can run four Gismos concurrently. Each virtual space on the four Gismos is shared by users

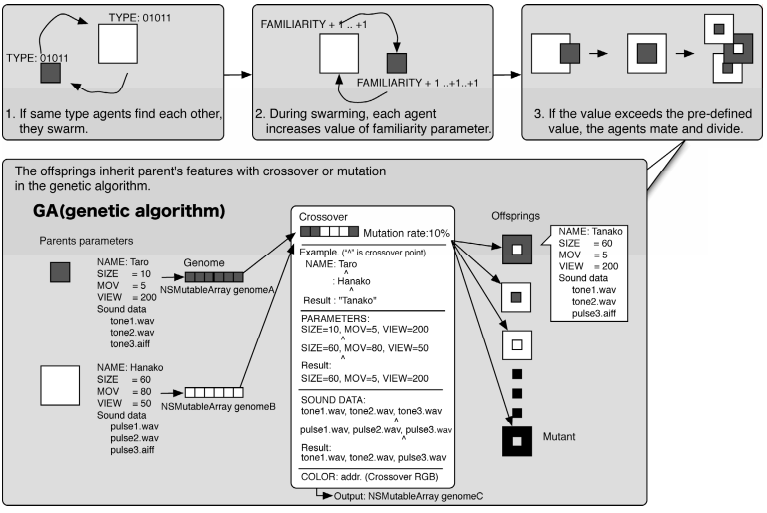


Fig. 6. Gismo mating implementation diagram

with the peer-to-peer communication. Consequently, encountered both side agents can interact as well as local agents.

5 Conclusions

The application for composition with multi agent system is characterized as follows.

Firstly, shift from the old model that depends on concept and music theory to the new model that expands of concepts and theory in the interactions between agents' and composer. Secondly**,** the composition style that operates relationships between agents by manipulating parameters and behaviours of them. Lastly**,** shift from the static components such as score, players and instruments to the interaction of sounds by the medium of agents.

Given this factor, Gismo adopts the method that builds up its entire structure in the feedback model, which repeats these three processes, designing of components, interactions between agents and discovering of emergences in interactions by the users. Eventually, the agent-based composition changes the established composition system into the more dynamic and flexible system.

References

[1] Joshua M. Epstein, Robert Axtell. *Growing Artificial Societies: Social Science from the Bottom up.* Massachusetts and London: The MIT Press, 1996.

[2] Stuart Kauffman. *At Home in the Universe: The Search for Laws of Self-Organization and Complexity.* New York: Oxford University Press, 1997.

[3] Tim Blackwell. *Swarm Music: Improvised Music with Multi-Swarms.* In Proc AISB03 Artificial Intelligence and Creativity in Arts and Science, pp.41-49 April. 2003.

[4] Apple Computer, Inc. “Xcode”. Apple Developer Connection. online, available from http://developer.apple.com/tools/xcode/ , accessed 2004-12-12.
[5] Curtis Roads. *Microsound.* Cambridge, Massachusetts and London: The MIT Press, 2001.
[6] Craig W. Reynolds, *Flocks, Herds, and Schools:A Distributed Behavioral Model*. in Proc. SIGGRAPH87, July. 1987, pp. 25-34.
[7] Matthew Wright, Adrian Freed. Open SoundControl: A New Protocol for Communicating with Sound Synthesizers. In Proc ICMC97, pp. 101-104 September. 1997.

Variable-Size Memory Evolutionary Algorithm to Deal with Dynamic Environments

Anabela Simões[1,2] and Ernesto Costa[2]

[1] Dept. of Informatics Engineering, ISEC – Coimbra Polytechnic, R. Pedro Nunes
Quinta da Nora, 3030-199 Coimbra - Portugal
[2] CISUC, University of Coimbra, Polo II, 3030-290 Coimbra - Portugal
abs@isec.pt, ernesto@dei.uc.pt

Abstract. When dealing with dynamic environments two major aspects must be considered in order to improve the algorithms' adaptability to changes: diversity and memory. In this paper we propose and study a new evolutionary algorithm that combines two populations, one playing the role of memory, with a biological inspired recombination operator to promote and maintain diversity. The size of the memory mechanism may vary along time. The size of the (usual) search population may also change in such a way that the sum of the individuals in the two populations does not exceed an established limit. The two populations have minimum and maximum sizes allowed that change according to the stage of the evolutionary process: if an alteration is detected in the environment, the search population increases its size in order to readapt quickly to the new conditions. When it is time to update memory, its size is increased if necessary. A genetic operator, inspired by the biological process of conjugation, is proposed and combined with this memory scheme. Our ideas were tested under different dynamics and compared with other approaches on two benchmark problems. The obtained results show the efficacy, efficiency and robustness of the investigated algorithm.

1 Introduction

Evolutionary Algorithms (EAs) have been used with success in a wide area of applications. Traditionally, EAs are well suited to solve problems where the environment is static. The generational process of evolution often leads the EA to the best solution. However, most of real-world applications are dynamic and the algorithms used to solve them must be able to adapt to the new circumstances.

For this type of optimization, an effective EA must be able to deal with the changes, detecting and reacting rapidly when they occur. Classical EAs are not suited for this kind of problems, since they have the tendency to prematurely converge to a solution and, when the conditions of the environment change, the population has all individuals usually concentrated in a specific point of the search space. So, it takes some time for the population to readapt and move towards the new solution.

To deal with these limitations, some improvements have been proposed as extensions of the classical EA. These improvements include 1) maintaining diversity using several strategies [3, 5, 7, 9, 14], 2) using memory schemes either implicit

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 617–626, 2007.
© Springer-Verlag Berlin Heidelberg 2007

[4, 10, 12] or explicit [1, 14, 19, 21] and 3) using multi-populations [2, 17]. Recent works successfully studied the combination of memory schemes and mechanisms for promoting and maintaining diversity [14, 18].

In this paper we are also interested in studying the combination of these two important issues. We propose an EA with a search population and a memory population whose size may change. Both search and memory populations have minimum and maximum values and the global number of individuals in the two populations cannot go beyond an established value. The memory is updated from time to time and its size changes whenever it is necessary and possible. If the maximum size is reached, the memory is cleaned to make room for new individuals. When a change occurs, the best individuals from the memory are selected and inserted in the population.

Besides this memory framework, we introduce a new way of using the biologically inspired conjugation operator, which improves the quality of the solutions. Conjugation will be used as the main recombination operator. This mechanism is applied after the mating pool is selected. The best individuals of the pool will transfer part of their genetic material to the worst individuals of the pool. After that, the new individuals are merged with the old population to form the next population.

2 Memory and Diversity: Relevant Background

The most known approach using redundant representations is to use diploid instead of haploid chromosomes. This idea was suggested by [4] as an extension of the standard GA. Later, other authors investigated the idea of diploidy, particularly its use in the context of dynamic environments [10, 12] .

When using explicit memory the main goal is to store useful information (good solutions) about the current environment and reuse them when a change occurs. It can also permit the population to move to a different area in the landscape in one step, which would not be possible with common genetic operators [2]. Different approaches have been proposed in the literature. For instance, [1, 14, 19, 21].

Several techniques have been proposed in order to preserve the diversity in a population. The most popular techniques to promote populations' diversity are hypermutation [3, 9], random immigrants [5], tag bits [7] or alternative genetic operators [14]. Their aim is to maintain or increase population's diversity in an EA allowing a quick reaction when modifications are detected. There are some works where these two issues – diversity and memory – are combined within the same algorithm claiming improved performance [14, 18].

3 Variable-Size Memory Evolutionary Algorithm

We propose a new EA called VMEA – Variable-size Memory Evolutionary Algorithm, comprising two populations: the main population searches the best solution and evolves as usual through selection, crossover and mutation. The main difference is that its size can change between two bounds: POP_MIN and POP_MAX.

The second population plays the role of a memory, where the best individuals of the population in several points of the generational process are stored. Its size also changes between off-line established limits MEM_MIN and MEM_MAX. The sum of the two populations cannot go beyond a certain limit (TOTAL_MAX). The individuals of the memory are aged: their age starts at zero, being increased by one at every generation. If an individual is selected to the population when a change is detected, an extra value is added to its age. Oldest individuals are those who stay longer in memory and/or contributed to adaptability of the population in an environmental change. If individuals reach a LIMIT_AGE their ages are reset to zero. The age of the individuals in the memory is used to select which individual to choose to withdraw when memory is full (or the sum of the size of the two populations is equal to the permitted limit). The memory is updated from time to time and if the established limits are not reached, the best individual of the current population is stored. If there is no room to save this new solution, we first clean the memory removing the individuals with the same genotype. If no individual was deleted through this process of cleaning, then the best individual of the current population, if is fittest than the one with lowest age present in the memory, replaces it.

The memory is evaluated every generation and a change is detected if at least one individual in the memory changes its fitness (as it was suggested in [1] and [21]) and it is updated at time T_M=t + *rand*(5,10), the same way as in [21].

If an environmental modification is detected, the best individual of the memory is introduced into the population. In the case of either the population's size or the sum of the two populations reach the allowed maximum, the best individual in memory replaces the worst one in the current population. A complete description of the algorithm can be consulted in [16].

4 Promoting Diversity in the Search Population

Traditionally EAs use crossover as the main genetic operator. In the past other biologically inspired operators have been proposed and tested with some degree of success. These new genetic operators were applied either in static [6, 13], or dynamic environments [14, 18].

In biology, bacterial conjugation is the transfer of genetic material between bacteria through cell-to-cell contact. Sometimes bacterial conjugation is regarded as the bacterial equivalent of sexual reproduction or mating, but in fact, it is merely the transfer of genetic information from a donor to a recipient cell [11].

Computational conjugation was introduced independently by Harvey and Smith. Smith [17] proposed an implementation of this operator, called simple conjugation: the donor and the recipient were chosen randomly, transferring the genetic material between two random points. Harvey [6] introduced a tournament based conjugation: two parents are selected on a random basis, and then the winner of the tournament becomes the donor and the loser the recipient of the genetic material. That way, the conjugation operator can be applied repeatedly by different donors to a single recipient.

In this paper conjugation is applied differently. We perform conjugation involving the individuals *selected to the mating pool*, using the idea of donor-recipient genetic

transfer. As it happens in biology, the donor individuals give genetic material to the recipient ones. After selecting the individuals to mate, using the established selection method, they are divided into two groups: the $\boldsymbol{n}/2$ best individuals become the 'donor', the remaining become the 'recipient' ($\boldsymbol{n}$ is the current size of the population). Then, the i^{th} donor transfers part of its genetic material to the i^{th} recipient (i=1, …n/2). This injection is controlled by two points randomly chosen. The donor remains unchanged. Following that, all offspring created by this process are joined with the donor individuals and they become the next population of size **n**. A complete explanation of conjugation is given in [16].

5 Experimental Study

5.1 Dynamic Test Environments

We selected two benchmark problems to test our VMEA so we can compare it with other algorithms easier: the knapsack problem (100 items) and *OneMax* problem (300 bits). In this paper, due to the restrictions of space, we only show partial results. A detailed set of results can be consulted in [16].

Knapsack Problem
The *knapsack* problem is a NP-complete combinatorial optimization problem often used as benchmark. It consists in selecting a number of items to a knapsack with limited capacity. Each item has a value (v_i) and a weight (w_i) and the objective is to choose the items that maximize the total value, without exceeding the capacity of the bag. We used a knapsack problem with 100 items using strongly correlated sets of randomly generated data [8, 20]. The fitness of an individual is equal to the sum of the values of the selected items, if the weight limit is not reached. If too many items are selected, then the fitness is penalized in order to ensure that invalid individuals are distinguished from the valid ones.

OneMax problem
The *OneMax* problem aims to maximize the number of ones in a binary string. So the fitness of an individual consists in the number of ones present in the binary string. This problem has a unique solution. In our experiments we used individuals of length 300.

Three kinds of environments were created for each of these two base problems, using Yang's Dynamic Optimization Problems (DOP) generator: cyclic, cyclic with noise and random. The environment was changed every $\boldsymbol{r}$ generations ($\boldsymbol{r}$ = 10, 50, 100 and 200) and the ratio $\boldsymbol{\rho}$ was set to different values in order to test different levels of change: 0.1 (a light shifting) 0.2, 0.5, 1.0 (severe change). In order to study the behaviour of the algorithms in randomly changing environments we also set $\boldsymbol{\rho}$ to a uniformly randomly generated value in the interval [0.01, 0.99] (called by ***rnd***).

Details about Yang's DOP generator can be found in [19]. With this generator it is possible to construct different dynamic environments from any binary-encoded stationary function using the bitwise exclusive-or (XOR) operator. The basic idea of the generator can be described as follows: when evaluating an individual **x** in the population, first we perform the operation $\mathbf{x} \oplus \mathbf{M}$ where $\oplus$ is the bitwise XOR

operator and **M** a binary mask previously generated. Then, the resulting individual is evaluated to obtain its fitness value. If a change happens at generation **t**, then we have $f(x, t+1) = f(x \oplus M)$. Using the DOP generator the characteristics of the change are controlled by two parameters: the speed of the change, $\boldsymbol{r}$, that is the number of generations between two changes, and the magnitude of the change, $\boldsymbol{\rho}$, that consists in the ratio of ones in the mask **M**. The more ones in the mask the more severe is the change. The DOP generator also allows constructing problems where the changes can be cyclic, cyclic with noise or non-cyclic. In the first case, several masks are generated according to the $\boldsymbol{\rho}$ parameter and are consecutively applied when a change occurs. It is thus possible that previous environments reappear later. In the second case noise is added by mutating some bits in the mask with a small probability. In the third case, the mask applied to the individuals is always randomly generated every time we change the environment.

5.2 Experimental Setup

Algorithms' parameters

To compare our approach we used two other algorithms: the random immigrants algorithm [5] and the memory-enhanced GA (MEGA) studied in [21]. VMEA was tested using conjugation (VMEA-Cj) and uniform crossover (VMEA-Cx), in order to make conclusions about the efficiency of the proposed genetic operator in changing problems. For all the algorithms, parameters were set as follows: generational replacement with elitism of size one, tournament selection with tournament of size two, uniform crossover with probability p_c=0.7 (the same probability was used with conjugation) and mutation applied with probability p_m=0.01. The population size for VMEA, RIGA and MEGA was set to 120 individuals. In MEGA, 10 individuals were used as memory, which is updated according the description given in [21]. The ratio of immigrants used in RIGA was 0.1. The mutation ratio used for noisy environments was 0.05. In VMEA the memory size varied between 10 and 50 individuals. However, the total of individuals in the two populations could not surpass 120. The age limit for the individuals in memory was set to $\boldsymbol{G/2}$, where $\boldsymbol{G}$ is the total number of generations.

For each experiment of an algorithm, 30 runs were executed and the number of environmental changes was 100 with $\boldsymbol{r}$ =10 (1000 generations), 40 with $\boldsymbol{r}$ =50 (2000 generations) and 20 with $\boldsymbol{r}$=100 and 200 (2000 and 4000 generations, respectively). The overall performance used to compare the algorithms was the best-of-generation fitness averaged over 30 independent runs, executed with the same random seeds:

$$Foverall = \frac{1}{G}\sum_{i=1}^{G}\left[\frac{1}{N}\sum_{j=1}^{N}F_{bestij}\right] \quad (1)$$

G=number of generations, N=number of runs.

5.3 Experimental Results

The experimental results carried out to assess the efficiency of our algorithm shows that VMEA outperformed the other two approaches. The statistical results comparing the algorithms are reported in tables 1 and 2. We used paired one-tailed t-test at a 0.01 level of significance. The notation used in tables 1 and 2, to compare each pair of algorithms is "+", "-", "++" or "--", when the first algorithm is *better* than, *worse*

than, *significantly better* than, or *significantly worse* than the second algorithm. Fig. 1 plots the average of the best-of-generation fitness obtained in the knapsack problem. Fig. 2 and 3 show some examples of the algorithms' behaviour during the generations.

Table 1. The t-test results of comparing the different algorithms (*knapsack* problem)

		CYCLIC					CYCLIC WITH NOISE					NON CYCLIC				
Statistical significance	r, ρ→	0.1	0.2	0.5	1	rnd	0.1	0.2	0.5	1	rnd	0.1	0.2	0.5	1	rnd
VMEA Cx - RIGA	10	++	++	++	++	++	--	++	++	++	--	++	-	--	++	--
VMEA Cj - RIGA		++	++	++	++	++	--	++	++	++	--	++	--	--	++	--
VMEA Cx - MEGA		++	++	++	++	++	+	++	++	++	++	++	+	++	++	++
VMEA Cj - MEGA		++	++	++	++	++	+	++	++	++	-	++	--	--	++	++
VMEA Cj - VMEA Cx		++	++	++	++	++	-	-	++	++	-	++	--	--	++	--
VMEA Cx - RIGA	50	++	++	++	++	++	++	++	++	++	++	++	++	+	++	++
VMEA Cj - RIGA		++	++	++	++	++	++	++	++	++	++	++	++	-	++	++
VMEA Cx - MEGA		++	++	++	++	++	++	++	++	++	++	++	++	++	++	++
VMEA Cj - MEGA		++	++	++	++	++	++	++	++	++	++	++	++	++	++	++
VMEA Cj - VMEA Cx		++	++	++	++	++	++	++	++	++	++	++	-	--	++	++
VMEA Cx - RIGA	100	++	++	++	++	++	++	++	++	++	++	++	++	++	++	++
VMEA Cj - RIGA		++	++	++	++	++	++	++	++	++	++	++	++	++	++	++
VMEA Cx - MEGA		++	++	++	++	++	++	++	++	++	++	++	++	++	++	++
VMEA Cj - MEGA		++	++	++	++	++	++	++	++	++	++	++	++	++	++	++
VMEA Cj - VMEA Cx		++	++	++	++	++	++	++	++	++	++	++	++	--	++	++
VMEA Cx - RIGA	200	++	++	++	++	++	++	++	++	++	++	++	++	++	++	++
VMEA Cj - RIGA		++	++	++	++	++	++	++	++	++	++	++	++	++	++	++
VMEA Cx - MEGA		++	++	++	++	++	++	++	++	++	++	++	++	++	++	++
VMEA Cj - MEGA		++	++	++	++	++	++	++	++	++	++	++	++	++	++	++
VMEA Cj - VMEA Cx		++	++	++	++	++	++	++	++	++	++	++	++	+	++	++

Table 2. The t-test results of comparing the different algorithms (*OneMax* problem)

		CYCLIC					CYCLIC WITH NOISE					NON CYCLIC				
Statistical significance	r, ρ→	0.1	0.2	0.5	1	rnd	0.1	0.2	0.5	1	rnd	0.1	0.2	0.5	1	rnd
VMEA Cx - RIGA	10	++	++	++	++	++	++	++	++	++	++	++	++	+	++	+
VMEA Cj - RIGA		++	++	++	++	++	--	--	--	--	--	--	--	--	++	--
VMEA Cx - MEGA		++	++	++	++	++	++	++	++	++	++	++	++	++	++	++
VMEA Cj - MEGA		++	++	++	++	++	--	--	--	--	--	--	--	--	++	--
VMEA Cj - VMEA Cx		--	--	--	--	--	--	--	--	--	--	--	--	--	++	--
VMEA Cx - RIGA	50	++	++	++	++	++	++	++	++	++	++	++	++	++	++	++
VMEA Cj - RIGA		++	++	++	++	++	++	++	++	++	+	++	--	--	++	--
VMEA Cx - MEGA		++	++	++	++	++	++	++	++	++	++	++	++	++	++	++
VMEA Cj - MEGA		--	++	++	++	--	--	--	--	--	--	--	--	--	++	--
VMEA Cj - VMEA Cx		--	--	++	++	--	--	--	--	--	--	--	--	--	++	--
VMEA Cx - RIGA	100	++	++	++	++	++	++	++	++	++	++	++	++	++	++	++
VMEA Cj - RIGA		++	++	++	++	++	++	++	++	++	++	++	++	--	++	++
VMEA Cx - MEGA		++	++	++	++	++	++	++	++	++	++	++	++	++	++	++
VMEA Cj - MEGA		--	++	++	++	--	--	--	--	-	--	--	--	--	++	--
VMEA Cj - VMEA Cx		--	--	++	++	--	--	--	--	--	--	--	--	--	++	--
VMEA Cx - RIGA	200	++	++	++	++	++	++	++	++	++	++	++	++	++	++	++
VMEA Cj - RIGA		++	++	++	++	++	++	++	++	++	++	++	++	++	++	++
VMEA Cx - MEGA		++	++	++	++	++	++	++	++	++	++	++	++	++	++	++
VMEA Cj - MEGA		++	++	++	++	++	++	++	++	++	++	++	-	--	++	-
VMEA Cj - VMEA Cx		++	++	++	++	++	--	--	--	--	--	++	--	--	+	--

In *cyclic environments*, our approach obtained at all times the best solutions. As expected, RIGA had the worst performance, obviously because it doesn't use any memory mechanism. Comparing VMEA and MEGA, we can conclude that the mechanism we introduced performs very well in cyclic environments. Also, conjugation shows a better performance in cyclic environment, obtaining almost all the times the best results. Decreasing the ratio of change, the effect of memory is not so visible. In fact, both memory algorithms, VMEA and MEGA, need some time to readapt when a change happens. This is because with small changes in the XOR mask, when a repeated state reappears, memory has already lost the useful information previously stored.

For *cyclic with noise and random environments* the results were in general very poor. The four algorithms didn't achieve high results as in the case of cyclic

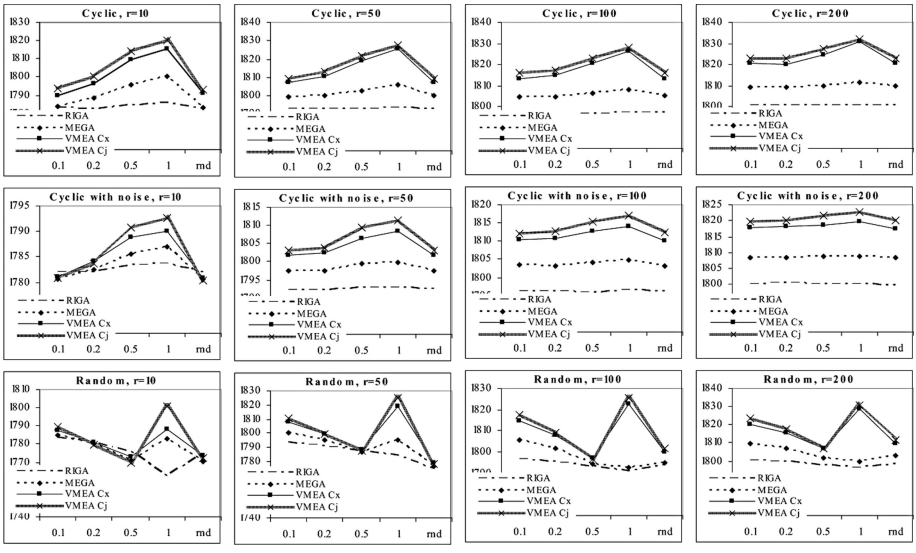


Fig. 1. Global results for the *knapsack* problem

environments. Nevertheless, VMEA obtained, in most cases, the best results, as we can see in Tables 1 and 2.

In *cyclic with noise environments*, algorithms behave in similar way: after a change they need some time to readapt and find a best solution. Memory improves the algorithm (VMEA-Cx achieves, in general, the highest scores), but its effect is not as obvious as in cyclic environment. Fig. 2 shows the evolutionary behavior of the algorithms for ρ=1 and ρ=0.1, with r=10 and r=200.

In *random environments*, with high ratio changes (ρ=1), VMEA achieved very good results. The memory allows the algorithm to continuously improve its performance. The good performance reduced as we decrease the change ratio. The new environment is slightly different from the previous one, but repeated states appear after a long time on and so memory has already lost the related information. Fig. 3 shows the behavior of the algorithms in noisy and random environments with ρ=1. For random change ratio (ρ = *rnd)* we observed a degradation of the results. In this case, RIGA and the other two memory-based algorithms performed in a very similar way: after a change in the environment the best-of generation falls for lower values and it is required some time for the algorithms to start evolving again. Even so, VMEA, typically arise the best marks.

The VMEA algorithm combined with the conjugation operator performed better in the knapsack problem. In fact, VMEA-Cj was, usually, the best algorithm. The same was not observed in the *OneMax* problem. In this benchmark problem, VMEA-Cx obtained the highest marks and VMEA with conjugation performed better in cyclic environments with larger change periods.

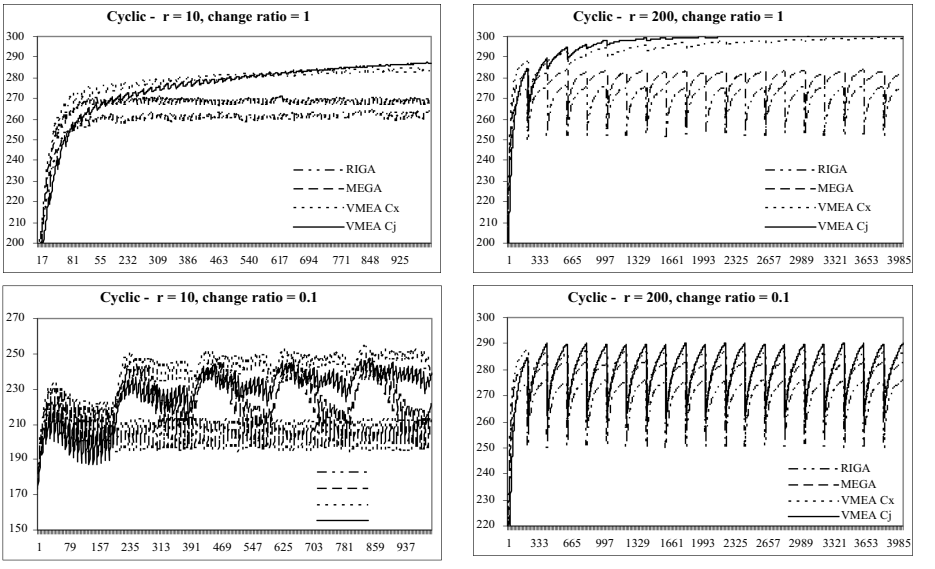


Fig. 2. Dynamic behavior of the algorithms in cyclic environments (*OneMax* problem)

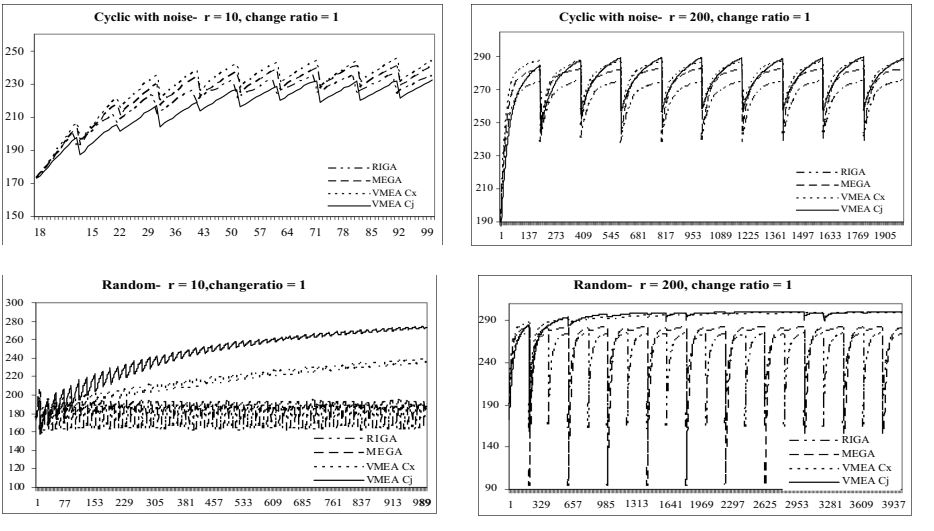


Fig. 3. Dynamic behavior in cyclic with noise and random environments (*OneMax* problem)

5.4 Memory and Population Sizes

The restrictions we impose when we increase the size of the memory and search populations imply that memory tends to grow until its maximum is achieved and so population is 'penalized' because we run out of resources. This happens because, when the established limits are attained, we only increase the population size when

there is room for at least one more individual and this is possible only if some individuals of the memory have been cleaned. After the maximum value for the memory is reached, the process of deleting individuals with the same genotype from the memory allows periodical increases in the population size. Fig. 4 shows a representative graphic of the evolution of the populations' size.

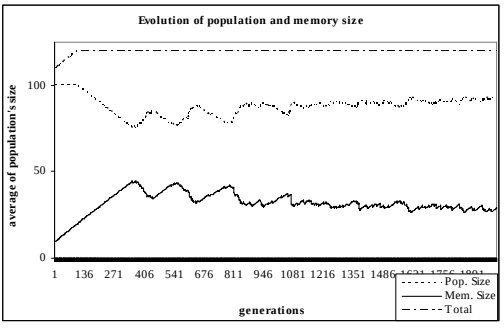


Fig. 4. Evolution of the population's and memory's size

6 Conclusions

In this paper we proposed an EA with memory of variable size to deal with dynamic environments. Additionally, we introduced a different biological operator to test its efficiency in the promotion of diversity. The investigated algorithm, called VMEA, was tested and compared with other approaches in different dynamic environments: cyclic, cyclic with noise and random. From the obtained results we can conclude that VMEA is very efficient. The best results were observed in cyclic environments: the greater the change ratio, the better the performance. For small change ratios, besides the change is not so severe, there are more different states reappearing in the environment. We can also conclude that the combination of the variable memory scheme and the conjugation operator increases the performance of the algorithm, mainly in cyclic environments. Finally, we must stress that for the implemented and compared algorithms, VMEA predominantly achieved the best results.

References

1. J. Branke. Memory Enhanced Evolutionary Algorithms for Changing Optimization Problems. *Proc. of the 1999 IEEE Congress on Evolutionary Computation*, pp. 1875-1882, 1999.
2. J. Branke. *Evolutionary Optimization in Dynamic Environments*. Norwell MA: Kluwer, 2001.
3. J. Branke, T. Kaußler, C. Schmidt and H. Schmeck. A Multi-Population Approach to Dynamic Optimization Problems. *Adaptive Computing in Design and Manufacture (ACDM 2000)*, pp. 299-308, 2000.

4. H. Cobb. An Investigation into the Use of Hypermutation as an Adaptive Operator in Genetic Algorithms Having Continuous, Time-Dependent Non-Stationary Environments. *Technical Report* AIC-90-001, 1990.
5. D. E. Goldberg and R. E. Smith. Nonstationary Function Optimization using Genetic Algorithms with Dominance and Diploidy. *Proc. of the 2nd Int. Conference. on Genetic Algorithms*, pp. 59-68, 1987.
6. J. J. Grefenstette. Genetic Algorithms for Changing Environments. *Proc. of the 2nd Int. Conference Parallel Problem Solving from Nature 2*, pp. 137-144, North-Holland, 1992.
7. I. Harvey. The Microbial Genetic Algorithm. *Unpublished*, 1996.
8. W. Liles and K. De Jong. The Usefulness of Tag Bits in Changing Environments. *Proceedings of 1999 IEEE Congress on Evolutionary Computation*, pp. 2054-2060, 1999.
9. Z. Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs*. 3rd Edition Springer-Verlag, 1999.
10. R. W. Morrison and K. De Jong. Triggered Hypermutation Revisited. *Proc. of the 1999 IEEE Congress on Evolutionary Computation*, pp. 1025-1032, 1999.
11. K. P. Ng and K. C. Wong. A New Diploid Scheme and Dominance Change Mechanism for Non-stationary Function Optimization. *Proc. of the 6th Int. Conference on Genetic Algorithms*, pp. 159-166, 1995.
12. P. J. Russell. *Genetics*. 5th edition, Addison-Wesley, 1998.
13. A. Sima Uyar and A. Emre Harmanci. A New Population Based Adaptive Domination Change Mechanism for Diploid Genetic Algorithms in Dynamic Environments. *Soft Computing*, vol. 9, pp. 803-814, 2005.
14. A. Simões and E. Costa. Transposition: A Biologically Inspired Mechanism to Use with Genetic Algorithms. *Proc. of the 4th Int. Conf. on Artificial Neural Networks*, pp. 612- 19, 1999.
15. A. Simões and E. Costa. An Immune System-Based Genetic Algorithm to Deal with Dynamic Environments: Diversity and Memory. *Proc. of the 6th Int. Conference on Artificial Neural Networks*, pp. 168-174, 2003.
16. A. Simões and E. Costa. Variable-size Memory Evolutionary Algorithm to Deal with Dynamic Environments: an empirical study. *CISUC Technical Report*, TR 2006/004 - ISSN 0874-338X, November 2006.
17. P. Smith. Conjugation: A Bacterially Inspired Form of Genetic. *Late Breaking Papers at the Genetic Programming 1996 Conference*, 1996.
18. M. Wineberg and F. Oppacher. Enhancing GA's Ability to Cope with Dynamic Environments. *Proc. of the 2000 Genetic and Evolutionary Comput. Conf.*, pp. 3-10, 2000.
19. S. Yang. A Comparative Study of Immune System Based Genetic Algorithms in Dynamic Environments. *Proc. of the 2006 Genetic and Evolutionary Computation Conference*, pp. 1377-1384, 2006.
20. S. Yang. Associative Memory Scheme for Genetic Algorithms in Dynamic Environments. *Proc. of the EvoWorkshops 2006*, LNCS 3097, pp. 788-799, 2006.
21. S. Yang and X. Yao. Experimental Study on Population-Based Incremental Learning Algorithms for Dynamic Optimization problems. *Soft Computing*, vol. 9, nº 11, pp. 815-834, 2005.
22. S. Yang. Memory-Based Immigrants for Genetic Algorithms in Dynamic Environments. *Proc. of the 2005 Genetic and Evolutionary Computation Conference*, vol. 2, pp. 1115-1122, 2005.

Genetic Algorithms with Elitism-Based Immigrants for Changing Optimization Problems

Shengxiang Yang

Department of Computer Science, University of Leicester
University Road, Leicester LE1 7RH, United Kingdom
s.yang@mcs.le.ac.uk

Abstract. Addressing dynamic optimization problems has been a challenging task for the genetic algorithm community. Over the years, several approaches have been developed into genetic algorithms to enhance their performance in dynamic environments. One major approach is to maintain the diversity of the population, e.g., via random immigrants. This paper proposes an elitism-based immigrants scheme for genetic algorithms in dynamic environments. In the scheme, the elite from previous generation is used as the base to create immigrants via mutation to replace the worst individuals in the current population. This way, the introduced immigrants are more adapted to the changing environment. This paper also proposes a hybrid scheme that combines the elitism-based immigrants scheme with traditional random immigrants scheme to deal with significant changes. The experimental results show that the proposed elitism-based and hybrid immigrants schemes efficiently improve the performance of genetic algorithms in dynamic environments.

1 Introduction

Many real world problems are dynamic optimization problems (DOPs) where change may occur over time with respect to all aspects of the problem being solved. For example, the problem-specific fitness evaluation function and constraints, such as design variables and environmental conditions, may change over time. Addressing DOPs has been a challenging task for the genetic algorithm (GA) community due to their dynamic characteristics [6,11]. For stationary optimization problems, our goal is to develop GAs that can quickly and precisely locate the optima of the fitness landscape. However, for DOPs quickly and precisely locating the optimum solution(s) of a snapshot optimization problem is no longer the unique goal. Instead, tracking the changing environment becomes a more important issue. This challenges traditional GAs due to the convergence problem because once converged GAs cannot adapt well to the changing environment. Over the years, several approaches have been developed into GAs to address DOPs [3], such as diversity schemes [5,7,12], memory schemes [1,2,10,14], and multi-population and species approaches [4,9].

Among the approaches developed for GAs for DOPs, the random immigrants scheme has proved to be beneficial for many DOPs. It works by maintaining the

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 627–636, 2007.
© Springer-Verlag Berlin Heidelberg 2007

diversity of the population by replacing individuals from the population with randomly created individuals. In this paper, an *elitism-based immigrants* scheme is proposed and investigated for GAs in dynamic environments. In this scheme, the elite from previous generation is used as the base to create immigrants via mutation to replace the worst individuals in the current population. This way, the introduced immigrants are more adapted to the current environment than random immigrants. This paper also proposes a hybrid immigrants scheme that combines the elitism-based immigrants scheme and traditional random immigrants scheme in order to deal with significant changes.

Based on the dynamic problem generator proposed in [13,15], a series of dynamic test problems are constructed from several stationary functions and experimental study is carried out to compare the performance of several GA variants with different immigrants schemes. Based on the experimental results, we analyze the performance of GAs regarding the weakness and strength of immigrants schemes for GAs in dynamic environments. The experiment results show that the proposed elitism-based immigrants scheme and the hybrid immigrants scheme efficiently improves the performance of GAs in dynamic environments.

The rest of this paper is outlined as follows. The next section briefly reviews random immigrants for GAs in dynamic environments. Section 3 presents the proposed elitism-based and hybrid immigrants schemes for GAs in dynamic environments. Section 4 describes the dynamic test environments for this study. The experimental results and analysis are presented in Section 5. Finally, Section 6 concludes this paper with discussions on future work.

2 Random Immigrants for GAs in Dynamic Environments

The standard GA (SGA) maintains and evolves a population of candidate solutions through selection and variation. New populations are generated by first selecting relatively fitter individuals from the current population and then recombining them via crossover and mutation to create new off-spring. This process continues until some stop condition is met. Usually, with the iteration of the GA, the population will eventually converge to the optimum solution(s) due to the pressure of selection.

In stationary environments, convergence at a proper pace is really what we expect for GAs to locate the optimum solution(s) for many optimization problems. However, for DOPs, convergence usually becomes a big problem for GAs because changing environments usually require GAs to keep a certain population diversity level to maintain their adaptability. To address this problem, the random immigrants approach is a quite natural and simple way [5,7]. It was proposed by Grefenstette with the inspiration from the flux of immigrants that wander in and out of a population between two generations in nature. It maintains the diversity level of the population through replacing some individuals of the current population with random individuals, called *random immigrants*, every generation. As to which individuals in the population should be replaced, usually there are two strategies: replacing random individuals or replacing the worst ones [12]. In

```
begin
  t := 0 and initialize population P(0) randomly
  evaluate population P(0)
  repeat
    P'(t) := selectForReproduction(P(t))
    crossover(P'(t), p_c)      // p_c is the crossover probability
    mutate(P'(t), p_m)         // p_m is the mutation probability
    evaluate the interim population P'(t)

    // perform elitism-based immigration
    denote the elite in P(t-1) by E(t-1)
    generate r_ei × n immigrants by mutating E(t-1) with p_m^i
    evaluate these elitism-based immigrants

    if the hybrid scheme is used then      // for HIGA
      generate r_ri × n random immigrants
      evaluate these random immigrants

    replace the worst individuals in P'(t) with the generated immigrants
    P(t+1) := P'(t)
  until the termination condition is met      // e.g., t > t_max
end
```

Fig. 1. Pseudo-code for the elitism-based immigrants GA (EIGA) and the hybrid immigrants GA (HIGA)

order to avoid that random immigrants disrupt the ongoing search progress too much, especially during the period when the environment does not change, the ratio of the number of random immigrants to the population size is usually set to a small value, e.g., 0.2.

3 The Elitism-Based Immigrants Scheme

As discussed above, traditional random immigrants approach works by replacing random individuals into the population. This may increase the population diversity level and hence may benefit GA's performance in dynamic environments, especially when a change occurs. However, in a slowly changing environment, the introduced random immigrants may divert the searching force of the GA during each environment before a change occurs and hence may degrade the performance. On the other hand, if the environment only changes slightly in terms of severity of changes, random immigrants may not have any actual effect even when a change occurs because individuals in the previous environment may still be quite fit in the new environment.

Based on the above consideration, this paper proposes an immigrants approach, called *elitism-based immigrants*, for GAs to address DOPs. Fig. 1 shows the pseudo-code for the GA with the proposed elitism-based immigrants scheme,

denoted *EIGA* in this paper. Within EIGA, for each generation t, after the normal genetic operations (i.e., selection and recombination), the elite $E(t-1)$ from previous generation is used as the base to create immigrants. From $E(t-1)$, a set of $r_{ei} \times n$ individuals are iteratively generated by mutating $E(t-1)$ bitwise with a probability p_m^i, where n is the population size and r_{ei} is the ratio of the number of elitism-based immigrants to the population size. The generated individuals then act as immigrants and replace the worst individuals in the current population. It can be seen that the elitism-based immigrants scheme combines the idea of elitism with traditional random immigrants scheme. It uses the elite from previous population to guide the immigrants toward the current environment, which is expected to improve GA's performance in dynamic environments.

In order to address significant changes that a DOP may suffer, the elitism-based immigrants can be hybridized with traditional random immigrants scheme. The pseudo-code for the GA with the hybrid immigrants scheme, denoted *HIGA* in this paper, is also shown in Fig. 1. Within HIGA, in addition to the $r_{ei} \times n$ immigrants created from the elite of previous generation, $r_{ri} \times n$ immigrants are also randomly created, where r_{ri} is the ratio of the number of random immigrants to the population size. These two sets of immigrants will then replace the worst individuals in the current population.

4 Dynamic Test Environments

The DOP generator proposed in [13,15] can construct dynamic environments from any binary-encoded stationary function $f(\boldsymbol{x})$ ($\boldsymbol{x} \in \{0,1\}^l$) by a bitwise exclusive-or (XOR) operator. The environment is changed every τ generations. For each environmental period k, an XORing mask $\boldsymbol{M}(k)$ is incrementally generated as follows:

$$\boldsymbol{M}(k) = \boldsymbol{M}(k-1) \oplus \boldsymbol{T}(k) \tag{1}$$

where "$\oplus$" is the XOR operator and $\boldsymbol{T}(k)$ is an intermediate binary template randomly created with $\rho \times l$ ones for environmental period k. For the first period $k = 1$, $\boldsymbol{M}(1) = \boldsymbol{0}$. Then, the population at generation t is evaluated as below:

$$f(\boldsymbol{x}, t) = f(\boldsymbol{x} \oplus \boldsymbol{M}(k)) \tag{2}$$

where $k = \lceil t/\tau \rceil$ is the environmental index. With this generator, the parameter τ controls the change speed while $\rho \in (0.0, 1.0)$ controls the severity of changes. Bigger ρ means severer changes while smaller τ means faster changes.

In this paper, three 100-bit binary-encoded problems are selected as the stationary functions. The first one is the $OneMax$ function, which aims to maximize the number of ones in a chromosome. The second one, denoted *Royal Road* due to its similarity to the Royal Road function by Mitchell et. al [8], consists of 25 contiguous 4-bit building blocks. Each building block contributes 4 to the total fitness if all its four bits are set to one; otherwise, it contributes 0. The third problem is a 100-item 0-1 knapsack problem with the weight and profit of each

item randomly created in the range of $[1, 30]$ and the capacity of the knapsack set to half of the total weight of all items. The fitness of a feasible solution is the sum of the profits of the selected items. If a solution overfills the knapsack, its fitness is set to the difference between the total weight of all items and the weight of selected items, multiplied by a small factor 10^{-5} to make it in-competitive with those solutions that do not overfill the knapsack.

Dynamic test environments are constructed from the three stationary functions using the aforementioned XOR DOP generator with τ set to 10 and 50 and ρ set to 0.1, 0.2, 0.5, and 1.0 respectively. Totally, a series of 8 DOPs are constructed from each stationary function.

5 Experimental Study

5.1 Experimental Design

In the experiments, four GAs were investigated on the above constructed DOPs. They are the standard GA (SGA), traditional random immigrants GA (denoted RIGA), EIGA and HIGA. All GAs are set as follows: generational, uniform crossover with $p_c = 0.6$, flip mutation with $p_m = 0.01$, and fitness proportionate selection with elitism of size 1. In order to have fair comparisons among GAs, the population size and ratios of immigrants are set such that each GA has 120 fitness evaluations per generation as follows: the population size n is set to 120 for SGA and 100 for RIGA, EIGA and HIGA, the ratio r_{ei} is set to 0.2 for EIGA and 0.1 for HIGA, and r_{ri} is set to 0.2 for RIGA and 0.1 for HIGA. For EIGA and HIGA, p_m^i of bitwise mutating the elite for immigrants is set to 0.01.

For each GA on a DOP, 50 independent runs were executed with the same set of random seeds. For each run of a GA on a DOP, 200 environmental changes were allowed and the best-of-generation fitness was recorded every generation. The overall offline performance of a GA on a DOP is defined as the best-of-generation fitness averaged over the 50 runs and over the data gathering period, as formulated below:

$$\overline{F}_{BOG} = \frac{1}{G}\sum_{i=1}^{G}(\frac{1}{N}\sum_{j=1}^{N}F_{BOG_{ij}}) \tag{3}$$

where $G = 200 * \tau$ is the total number of generations for a run, $N = 50$ is the total runs, and $F_{BOG_{ij}}$ is the best-of-generation fitness of generation i of run j.

5.2 Experimental Results and Analysis

The experimental results of GAs on the DOPs are presented in Table 1. The statistical results of comparing GAs by one-tailed t-test with 98 degrees of freedom at a 0.05 level of significance are given in Table 2. In Table 2, the t-test result regarding Alg. 1 − Alg. 2 is shown as "$s+$", "$s-$", "$+$", or "$-$" when Alg. 1 is significantly better than, significantly worse than, insignificantly better than,

Table 1. Experimental results with respect to overall performance of GAs

Performance	*OneMax*				*Royal Road*				*Knapsack*			
$\tau = 10$, $\rho \Rightarrow$	0.1	0.2	0.5	1.0	0.1	0.2	0.5	1.0	0.1	0.2	0.5	1.0
SGA	74.0	69.5	64.6	62.0	45.5	36.0	27.1	40.1	1020.4	979.5	933.9	895.1
RIGA	74.4	71.0	66.5	63.8	45.4	36.5	28.3	39.1	1042.3	1000.1	945.6	908.8
EIGA	86.9	77.0	63.7	55.9	53.7	37.5	25.7	46.7	1110.2	1028.3	921.6	871.5
HIGA	82.7	75.3	67.3	63.5	48.3	37.3	28.2	43.4	1054.8	1007.5	946.8	907.7
$\tau = 50$, $\rho \Rightarrow$	0.1	0.2	0.5	1.0	0.1	0.2	0.5	1.0	0.1	0.2	0.5	1.0
SGA	83.2	79.4	72.4	65.3	67.7	58.7	44.8	41.5	1110.3	1077.4	1011.1	929.7
RIGA	81.9	78.9	75.2	73.8	69.0	59.2	47.0	40.8	1125.4	1095.2	1040.8	1007.6
EIGA	97.6	94.2	81.9	63.6	85.9	72.0	48.5	43.4	1228.6	1183.6	1068.9	923.9
HIGA	94.7	90.2	82.6	80.9	76.1	64.3	49.4	42.7	1149.7	1114.0	1049.4	1013.0

Table 2. The t-test results of comparing GAs on dynamic test problems

t-test Result	*OneMax*				*Royal Road*				*Knapsack*			
$\tau = 10$, $\rho \Rightarrow$	0.1	0.2	0.5	1.0	0.1	0.2	0.5	1.0	0.1	0.2	0.5	1.0
$RIGA - SGA$	$s+$	$s+$	$s+$	$s+$	$-$	$s+$	$s+$	$s-$	$s+$	$s+$	$s+$	$s+$
$EIGA - SGA$	$s+$	$s+$	$s-$	$s-$	$s+$	$s+$	$s-$	$s+$	$s+$	$s+$	$s-$	$s-$
$EIGA - RIGA$	$s+$	$s+$	$s-$	$s-$	$s+$	$s+$	$s-$	$s+$	$s+$	$s+$	$s-$	$s-$
$HIGA - SGA$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$
$HIGA - RIGA$	$s+$	$s+$	$s+$	$s-$	$s+$	$s+$	$s-$	$s+$	$s+$	$s+$	$s+$	$s-$
$HIGA - EIGA$	$s-$	$s-$	$s+$	$s+$	$s-$	$s-$	$s+$	$s-$	$s-$	$s-$	$s+$	$s+$
$\tau = 50$, $\rho \Rightarrow$	0.1	0.2	0.5	1.0	0.1	0.2	0.5	1.0	0.1	0.2	0.5	1.0
$RIGA - SGA$	$s-$	$s-$	$s+$	$s+$	$s+$	$s+$	$s+$	$s-$	$s+$	$s+$	$s+$	$s+$
$EIGA - SGA$	$s+$	$s+$	$s+$	$s-$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s-$
$EIGA - RIGA$	$s+$	$s+$	$s+$	$s-$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s-$
$HIGA - SGA$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$
$HIGA - RIGA$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$
$HIGA - EIGA$	$s-$	$s-$	$s+$	$s+$	$s-$	$s-$	$s+$	$s-$	$s-$	$s-$	$s-$	$s+$

or insignificantly worse than Alg. 2 respectively. The results are also plotted in Fig. 2. The dynamic behaviour of GAs for the first 10 environments is plotted with respect to best-of-generation fitness against generation on the DOPs with $\tau = 50$ and $\rho = 0.1$ and $\rho = 1.0$ in Fig. 3, where the data were averaged over 50 runs. From the tables and figures several results can be observed.

First, RIGA does significantly outperform SGA on most dynamic test problems, see the t-test results regarding $RIGA - SGA$ in Table 2. This result validates the benefit of introducing random immigrants for the GA for DOPs. However, on the *OneMax* problems with $\tau = 50$ and $\rho = 0.1$ and 0.2, RIGA is beaten by SGA. This confirms our prediction made in Section 3: when the environment changes slowly and slightly, random immigrants may not be beneficial.

Second, EIGA outperforms SGA and RIGA on most DOPs with $\tau = 50$ and on DOPs with $\tau = 10$ and $\rho = 0.1$ and 0.2, see the t-test results regarding $EIGA - SGA$ and $EIGA - RIGA$ in Table 2. This result confirms our expectation of

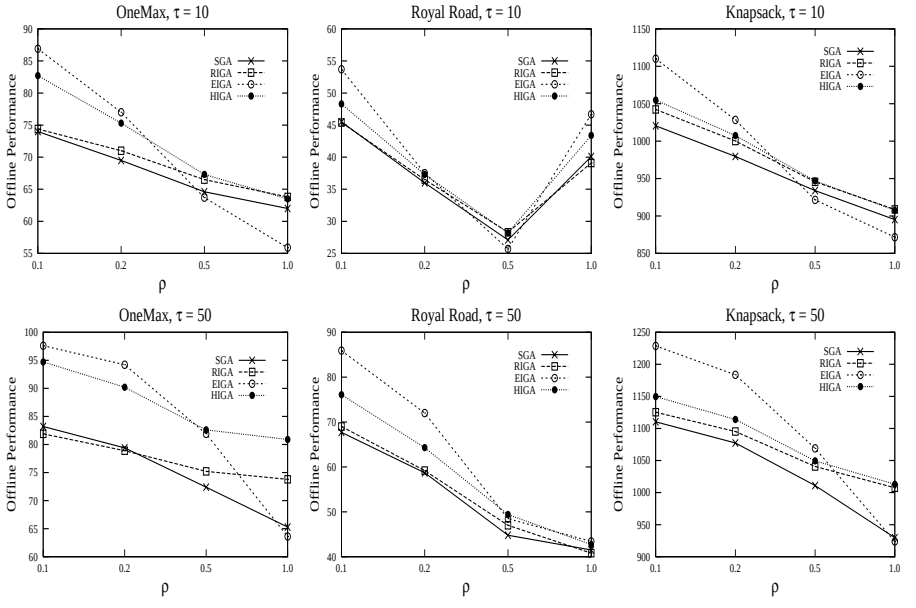


Fig. 2. Experimental results of GAs on the dynamic test problems

the elitism-based immigrants scheme for GAs in dynamic environments. When the environment changes slowly or slightly, it would be better to introduce immigrants guided toward the current environment via the elite. For example, see Fig. 3 for the dynamic behaviour of GAs on DOPs with $\tau = 50$ and $\rho = 0.1$. For each environment EIGA manages to maintain a much higher fitness level than SGA and RIGA.

When the environment changes significantly, e.g., $\rho = 1.0$, EIGA is beaten by SGA and RIGA on dynamic *OneMax* and *Knapsack* problems. The reason lies in that each time when the environment changes significantly, the elite from the previous generation may become significantly unfit in the newly changed environment and hence will guide the immigrants to unfit area. This can be observed from the sharp drop of the dynamic performance of EIGA on dynamic *OneMax* and *Knapsack* with $\rho = 1.0$ in Fig. 3.

Third, regarding the effect of the hybrid immigrants scheme for GAs, it can be seen that HIGA now outperforms SGA and RIGA on almost all DOPs, see the t-test results regarding $HIGA - SGA$ and $HIGA - RIGA$ in Table 2. This result shows the advantage of the hybrid immigrants scheme over no immigrants and random immigrants schemes. When comparing the performance of HIGA over EIGA, it can be seen that HIGA beats EIGA on DOPs with ρ set to bigger value 0.5 and 1.0 while is beaten by EIGA on DOPs with $\rho = 0.1$ and 0.2. The hybrid immigrants scheme improves the performance of HIGA over EIGA in significantly changing environments at the price of degrading the performance

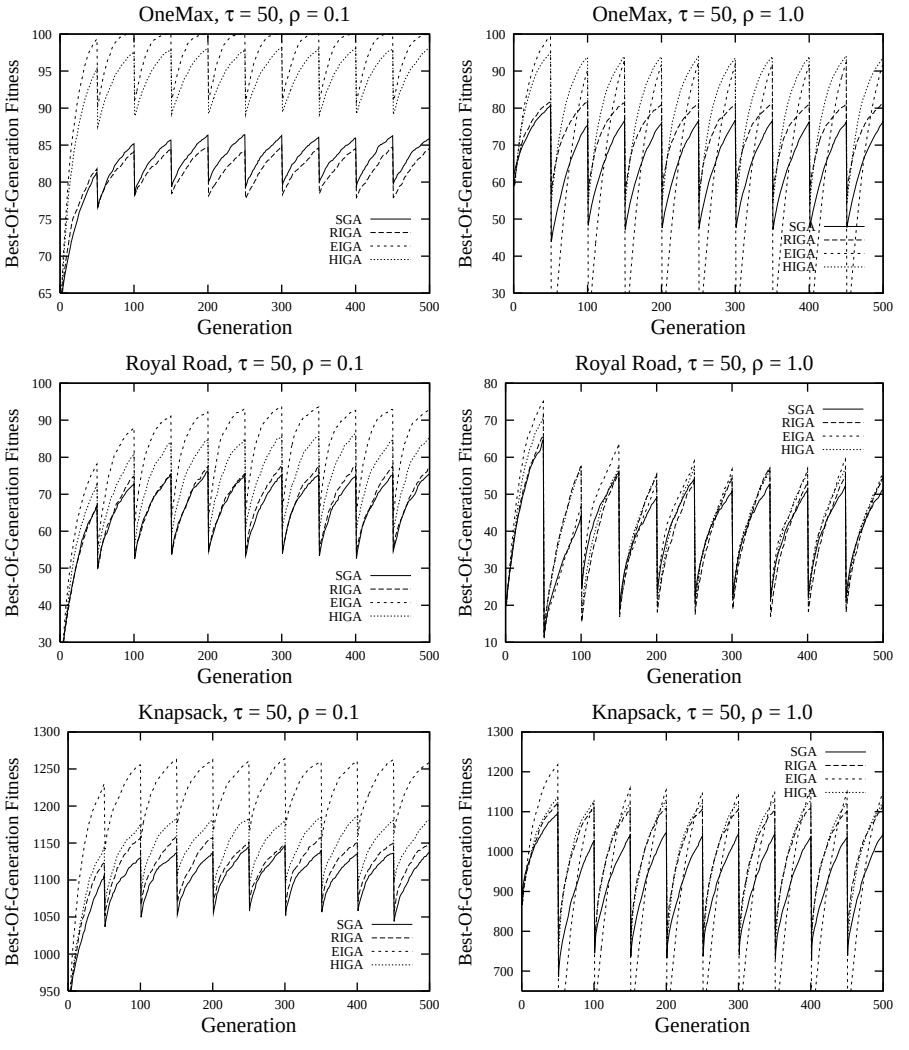


Fig. 3. Dynamic behaviour of GAs on DOPs with $\tau = 50$ for the first 10 environments

in slightly changing environments. This result can be more clearly observed from the dynamic performance of HIGA and EIGA in Fig. 3. The random immigrants added in HIGA prevent HIGA from climbing to the fitness level as high as EIGA does when the environment slightly changes with $\rho = 0.1$ while they also prevent the performance of HIGA from a sharp drop when the environment significantly changes with $\rho = 1.0$.

Finally, in order to understand the effect of investigated immigrants schemes on the population diversity, we recorded the diversity of the population every generation for each run of a GA on a DOP. The mean population diversity of

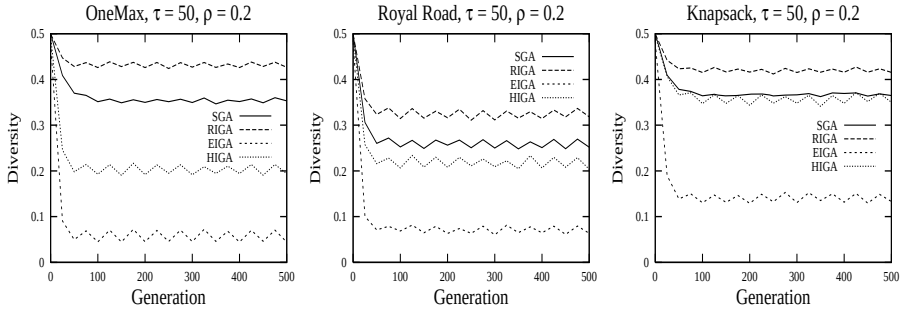


Fig. 4. Diversity dynamics of GAs on DOPs with $\tau = 50$ and $\rho = 0.2$ for the first 10 environments

a GA on a DOP at generation t over 50 runs is calculated according to the following formula:

$$\overline{Div}(t) = \frac{1}{50}\sum_{k=1}^{50}(\frac{1}{ln(n-1)}\sum_{i=1}^{n}\sum_{j\neq i}^{n} HD_{ij}(k,t)), \tag{4}$$

where $l = 100$ is the encoding length and $HD_{ij}(k,t)$ is the Hamming distance between the i-th and j-th individuals at generation t of the k-th run. The diversity dynamics over generations for GAs on DOPs with $\tau = 50$ and $\rho = 0.2$ is shown in Fig. 4. From Fig. 4, it can be seen that RIGA does maintain the highest diversity level in the population while EIGA maintains the lowest diversity level. This interesting result shows that approaches that aim at maintaining a high diversity level in the population, though usually useful, do not naturally achieve better performance than other approaches for GAs in dynamic environments.

6 Conclusions

The random immigrants scheme is one of several approaches developed into GAs to address DOPs. This paper proposes an elitism-based immigrants scheme for GAs in dynamic environments, where the elite from last generation is used as the base to create immigrants into the population via a normal bit flip mutation. This way, the introduced immigrants become more adapted to the current environment and hence more efficient in improving GA's performance. The elitism-based immigrants scheme can be combined with traditional random immigrants scheme to further improve the performance of GAs in dynamic environments.

From the experiment results on a series of dynamic problems, the following conclusions can be drawn. First, random immigrants are beneficial for most dynamic environments. Second, the proposed elitism-based immigrants scheme combines the working principles of random immigrants and elitism approaches and improves GA's performance for DOPs, especially in slowly or slightly changing environments. Third, the hybrid immigrants scheme seems a good choice for

GAs for DOPs. Finally, a high diversity level of the population does not always mean better performance of GAs in dynamic environments.

As relevant future work, it is interesting to compare the elitism-based and hybrid immigrants schems with other advanced diversity schemes, e.g., diversity and memory hybrid schemes [10,14], for GAs in dynamic environments. Another interesting work is to further integrate the idea of elitism and immigrants into other approaches, e.g., multi-population and speciation schemes [4,9], to develop advanced diversity schemes for GAs in dynamic environments.

References

1. C. N. Bendtsen and T. Krink. Dynamic memory model for non-stationary optimization. *Proc. of the 2002 Congress on Evol. Comput.*, pp. 145–150, 2002.
2. J. Branke. Memory enhanced evolutionary algorithms for changing optimization problems. *Proc. of the 1999 Congr. on Evol. Comput.*, vol. 3, pp. 1875–1882, 1999.
3. J. Branke. *Evolutionary Optimization in Dynamic Environments*. Kluwer Academic Publishers, 2002.
4. J. Branke, T. Kaußler, C. Schmidth, and H. Schmeck. A multi-population approach to dynamic optimization problems. *Proc. of the Adaptive Computing in Design and Manufacturing*, pp. 299–308, 2000.
5. H. G. Cobb and J. J. Grefenstette. Genetic algorithms for tracking changing environments. *Proc. of the 5th Int. Conf. on Genetic Algorithms*, pp. 523–530, 1993.
6. D. E. Goldberg and R. E. Smith. Nonstationary function optimization using genetic algorithms with dominance and diploidy. *Proc. of the 2nd Int. Conf. on Genetic Algorithms*, pp. 59–68, 1987.
7. J. J. Grefenstette. Genetic algorithms for changing environments. *Parallel Problem Solving from Nature II*, pp. 137–144, 1992.
8. M. Mitchell, S. Forrest and J. H. Holland. The royal road for genetic algorithms: fitness landscapes and GA performance. *Proc. of the 1st European Conf. on Artificial Life*, pp. 245–254, 1992.
9. D. Parrott and X. Li. Locating and tracking multiple dynamic optima by a particle swarm model using speciation. *IEEE Trans. on Evol. Comput.*, **10**(4): 444-458, 2006.
10. A. Simões and E. Costa. An immune system-based genetic algorithm to deal with dynamic environments: diversity and memory. *Proc. of the 6th Int. Conf. on Neural Networks and Genetic Algorithms*, pp. 168-174, 2003.
11. K. Trojanowski and Z. Michalewicz. Searching for optima in non-stationary environments. *Proc. of the 1999 Congress on Evol. Comput.*, pp. 1843–1850, 1999.
12. F. Vavak and T. C. Fogarty. A comparative study of steady state and generational genetic algorithms for use in nonstationary environments. *AISB Workshop on Evolutionary Computing, LNCS*, vol. 1143, pp. 297–304, 1996.
13. S. Yang. Non-stationary problem optimization using the primal-dual genetic algorithm. *Proc. of the 2003 Congr. on Evol. Comput.*, vol. 3, pp. 2246–2253, 2003.
14. S. Yang. Memory-based immigrants for genetic algorithms in dynamic environments. *Proc. of the 2005 Genetic and Evol. Comput. Conf.*, vol. 2, pp. 1115–1122, 2005.
15. S. Yang and X. Yao. Experimental study on population-based incremental learning algorithms for dynamic optimization problems. *Soft Computing*, **9**(11): 815-834, 2005.

Triggered Memory-Based Swarm Optimization in Dynamic Environments

Hongfeng Wang[1], Dingwei Wang[1], and Shengxiang Yang[2]

[1] School of Information Science and Engineering, Northeastern University
Shenyang 110004, P.R. China
{hfwang,dwwang}@mail.neu.edu.cn
[2] Department of Computer Science, University of Leicester
University Road, Leicester LE1 7RH, United Kingdom
s.yang@mcs.le.ac.uk

Abstract. In recent years, there has been an increasing concern from the evolutionary computation community on dynamic optimization problems since many real-world optimization problems are time-varying. In this paper, a triggered memory scheme is introduced into the particle swarm optimization to deal with dynamic environments. The triggered memory scheme enhances traditional memory scheme with a triggered memory generator. Experimental study over a benchmark dynamic problem shows that the triggered memory-based particle swarm optimization algorithm has stronger robustness and adaptability than traditional particle swarm optimization algorithms, both with and without traditional memory scheme, for dynamic optimization problems.

1 Introduction

In recent years, there has been an increasing concern from the evolutionary computation community on problem optimization in dynamic environments since many real-world problems are dynamic optimization problems (DOPs), where stochastic changes may occur regarding the optimization goal, the problem instance, or some restrictions. For DOPs, the goal of evolutionary algorithms (EAs) is no longer to find a satisfactory solution, but to track the trajectory of the moving optimum in the search space. This poses a great challenge to traditional EAs. To address this challenge, several approaches have been developed into EAs to improve their performance in dynamic environments [2,3,9,10,13,16]. A comprehensive survey can be found in [4].

The genetic algorithm (GA) was the first evolutionary computation approach used to explore DOPs. Recently, particle swarm optimization (PSO), as another evolutionary computation technique, has been applied to DOPs. In this paper, a triggered memory-based approach is introduced into the PSO to improve its performance in dynamic environments. The triggered memory scheme enhances traditional memory scheme with a triggered memory generator. Experimental study over a benchmark dynamic problem shows that the triggered memory scheme efficiently improves the performance of PSOs in dynamic environments.

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 637–646, 2007.
© Springer-Verlag Berlin Heidelberg 2007

The rest of this paper is outlined as follows. The next section briefly describes the PSO and surveys the literature on PSOs for DOPs. Section 3 proposes a variety of triggered memory-based methods for PSOs to handle dynamic environments. Then, the experimental settings and results are reported in Section 4 and Section 5 respectively. Finally, Section 6 concludes this paper with some suggestions for future work.

2 Particle Swarm Optimization in Dynamic Environments

Particle swarm optimization was first introduced by Kennedy and Eberhart [7,12]. PSO simulates the social behaviour among particles that "fly" through a solution space. Each particle accomplishes its own updating based on its current velocity and position, the best position seen so far by the particle, and the best position seen so far by the population (or by the local neighbourhood in the local version of PSOs. In this paper, only the global version is discussed). The behaviour of a particle can be described as follows:

$$v_i(t+1) = \omega v_i(t) + c_1\xi(p_i(t) - x_i(t)) + c_2\eta(p_g(t) - x_i(t)) \quad (1)$$

$$x_i(t+1) = x_i(t) + v_i(t+1) \quad (2)$$

where $v_i(t)$ and $x_i(t)$ represent the current velocity and position of particle i at time t respectively, $p_i(t)$ is the position of the best solution discovered so far by particle i, $p_g(t)$ is the position of the best solution found so far by all particles, ω is the inertia weight that controls the degree a particle's previous velocity will be kept, c_1 and c_2 are individual and social learning factors, and ξ and η are random numbers in the range [0, 1].

As a kind of robust optimization technique, PSO has been widely used for stationary optimization problems where the fitness landscape does not change during the course of the computation. Recently, the application area of PSOs has been extended to time-varying systems. When applied for DOPs, traditional PSOs face a big problem, that is, the whole population will eventually converge to a small area, from which it is very difficult for PSOs to jump out to follow the changes. This is a challenge to traditional PSOs for DOPs. Recently, researchers have developed a number of PSO approaches for DOPs, which are briefly reviewed below.

Eberhart and Shi [8] put forward the first work, where the PSO was investigated to track a single peak that varies spatially. Based on a parabolic function $f(\cdot) = x^2 + y^2 + z^2$, they observed that the tracking errors achieved by the standard PSO are several order of magnitude less than those achieved by comparable GA-based approaches.

Hu and Eberhart [11] introduced an adaptive PSO, which automatically tracks various changes in a dynamic system. They tested different environmental detection and re-randomization strategies, which effectively respond to a wide variety of changes, and reported and analyzed the experimental results on the parabolic function and Rosenbrock's benchmark function with various severities of environmental changes.

Parott and Li [14] investigated a PSO model for tracking multiple peaks in a continuously changing dynamic environment. Multiple parallel subpopulations were constructed by a form of speciation and encouraged to simultaneously tracking multiple peaks by preventing overcrowding at peaks in this model. The experiments in dynamic multimodal environments indicated that the technique was capable of tracking multiple changing peaks simultaneously.

A method of adapting PSO for dynamic environments was present by Carlisle and Dozier [6]. In their PSO, each particle can reset the record of its best position and avoid making direction and velocity decisions based on outdated information as the environment changed. Two resetting methods were examined and experimental results show that both were able to improve the performance of PSO in both static and dynamic environments.

Blackwell and Branke [1] proposed several new variants of PSOs specifically designed for non-stationary environments, where the single population PSO and charged PSO (CPSO) were extended by constructing interacting multi-swarms. In addition, a new multi-quantum swarm optimizer, which broadens the implicit atomic analogy of CPSO to a quantum model, was also introduced. Their experimental study on the Moving Peaks Benchmark problem indicates that the multi-swarm optimizers significantly outperform single population PSOs.

3 The Triggered Memory-Based PSO for DOPs

Among the approaches developed for EAs in dynamic environments, the memory scheme is a major approach that has proved beneficial for many DOPs [15]. In memory enhanced EAs, good individuals from the population can be stored into a memory at regular interval during the course of evolution and can be retrieved once a change occurs in the environment. Intuitively, when an optimum reappears in a previous location or nearby, the memory can remember that location and guide the population to move to that optimum. Memory can also help maintain the population diversity and adapt to environmental changes quickly because useful past information has been saved and can be reused. In this section, we discuss how a triggered memory mechanism can be applied to the PSO in order to make it suitable for dynamic problems.

A disadvantage of the memory schemes is that memory might mislead evolution and prevent the population from exploring new peaks in the search space, though it might be propitious to the exploitation of knowledge gained in the past. Intuitively, restarting evolution from scratch once a change in the environment has occurred will have a chance to find new peaks. However, it may be too time-consuming to reach the new optima. In [2], a tri-island model has been proposed for the memory-enhanced GA in dynamic environments and proved an efficient way to maintain the tradeoff between exploration and exploitation.

The idea of the tri-island memory model can be combined into the PSO for DOPs. For the memory-enhanced PSO with the tri-island model, the whole population is also divided into three parts: “explore”-population, memory and “exploit”-population, which are respectively used to explore the search space,

store good solutions, and exploit the memory in this paper. The memory is also used to detect the environmental changes. A change is detected to have occurred whenever the fitness of at least one solution in the memory has changed. Once an environmental change is detected, all individuals will be re-evaluated. In order to enforce exploration, the "explore"-population needs to be randomly re-initialized frequently. Thus, the re-initialization period, that is, when the "explore"-population should be re-initialized, becomes an important parameter. A simple scheme is to re-initialize the "explore"-population after every environmental change. However, there may be some problems for this scheme. For example, if the environment changes slowly, the population might always stay in a peak and might not jump out to search for other peaks for a long time.

In order to solve the above problem, we introduce a new triggered generator for the memory-based PSO, where the re-initialization of the "explore"-population will be immediately initiated once a peak has been found. Thus, the triggered generator can be more efficient in exploring the search space than the simple re-initialization method, especially when the environment does not change frequently. The next question is how to judge that a peak has been found by the "explore"-population. Here, we deem that a peak has been found if some restrictions on its performance are fulfilled. For the re-initialization conditions, we consider the following two alternatives:

First, compute the running average of the fitness of the best individuals over a period of five generations. If the increase degree of the running average of the best-of-generation fitness is less than a threshold $b1$, the re-initialization is started (this method will subsequently be termed *averfit*). The increase of the running average of the best-of-generation fitness is calculated as follows:

$$e(t) = \frac{\frac{1}{5}\sum_{i=0}^{4} f_b(t-i) - \frac{1}{5}\sum_{i=0}^{4} f_b(t-i-1)}{\frac{1}{5}\sum_{i=0}^{4} f_b(t-i)} = \frac{\sum_{i=0}^{4} f_b(t-i) - \sum_{i=0}^{4} f_b(t-i-1)}{\sum_{i=0}^{4} f_b(t-i)} \quad (3)$$

where $f_b(t)$ denotes the fitness of the best solution achieved in generation t.

Second, compute the Euclidean distance between the best individual and the worst individual in the "explore"-population. If the value is below a threshold $b2$, the population will be re-initialized. This method will subsequently be termed *maxdist* in this paper.

With respect to which individuals should be stored in the memory, we only consider the best individuals achieved by the "explore"-population. And since the memory space is fixed and limited, we need to consider which solutions in the memory should be replaced to make space for new ones. In this paper, in order to maintain the diversity of the memory, we apply the replacement strategy where the less fit solution of the two solutions that are closest to each other in the memory is removed from the memory.

For further questions as to when and how to use the memory, we propose two memory retrieval schemes: called *memory-based resetting* and *memory-based immigrants*. Both schemes happen at the same time when the "explore"-population

is re-initialized. The memory-based resetting scheme is to reset the record of the best global position using the memory, where all the solutions in the memory will be re-evaluated and the best one will be chosen as the new best global solution in the "exploit"-population if it is better than the old one. In this scheme, all particles together share a memory base that stores the best peaks achieved by the "explore"-population in the past. Therefore, the particles can adjust the direction of flying according to the memorial information. For the memory-based immigrants retrieval scheme, all the solutions in the memory are re-evaluated and injected into the "exploit"-population to replace the same amount of worst individuals in the "exploit"-population. Compared to the memory-based resetting scheme, this scheme seems more efficient since the solutions in the memory are explicitly immigrated to the population.

4 Experimental Settings

For the experiments, the Moving Peaks Benchmark by Branke [5] is used as the dynamic test problem. The base landscape of the moving peaks function consists of m peaks defined in the n-dimensional real space as follows:

$$F(\boldsymbol{x}, t) = \max_{i=1,\ldots,m} \frac{H_i(t)}{1 + W_i(t) \sum_{j=1}^{n} (x_j(t) - X_{ij}(t))^2} \tag{4}$$

where $W_i(t)$ and $H_i(t)$ are the height and width of peak i at time t respectively, and $X_{ij}(t)$ is the j-th element of the location of peak i at time t. Each peak can independently change its height and width and move its location around in the search space.

The parameter settings of the Moving Peaks Benchmark used in this paper correspond to Scenario 1 as specified on the benchmark website [5]. The test function has 5 peaks defined on a 5-dimensional real space. Every Δe generations, the height and width of each peak are changed by adding a random Gaussian variable and the location of each peak is moved by a shift vector $\boldsymbol{v_i}$ of fixed length s. More formally, a change of a single peak can be described as follows:

$$\begin{cases} \sigma \in N(0,1) \\ H_i(t) = H_i(t-1) + 7 \cdot \sigma \\ W_i(t) = W_i(t-1) + 0.01 \cdot \sigma \\ \boldsymbol{X}_i(t) = \boldsymbol{X}_i(t-1) + \boldsymbol{v}_i(t), \end{cases} \tag{5}$$

$$\boldsymbol{v}_i(t) = \frac{s}{|\boldsymbol{r} + \boldsymbol{v}_i(t-1)|}((1-\lambda)\boldsymbol{r} + \lambda \boldsymbol{v}_i(t-1)) \tag{6}$$

where the shift vector $\boldsymbol{v}_i(t)$ is a linear combination of a random vector $\boldsymbol{r}$ and the previous shift vector $\boldsymbol{v}_i(t-1)$ and is normalized to length s. The random vector $\boldsymbol{r}$ is created by drawing random numbers for each dimension and normalizing its length to s. Hence, the parameter s allows controlling the severity of changes and Δe determines the frequency of changes. The parameter λ allows controlling whether changes exhibit a trend (λ is always set to 0.5 in our experiments).

The experiments are designed to investigate the performance of different memory-based algorithms: a simple PSO model (SPSO) where each change is regarded as the arrival of a new optimization problem and is solved from scratch, a traditional memory-based PSO model (SMPSO) that is adapted from Branke's tri-island memory model for GAs [2], the triggered memory-based PSO models both with the resetting scheme (TMRPSO) and the immigrants scheme (TMIPSO). For all PSO models, the learning factors c_1 and c_2 are set to 2.0 and the inertia weight ω is initialized to 0.5, decreases linearly over the first 100 generations till 0.2, and then remains at 0.2 till the end of a run. The total number of particles is 50: the size of both the "explore"-population and "exploit"-population is set to 20. Unless stated otherwise, the size of memory is always 10.

To measure the performance for the algorithms, an offline performance function e^*, which is the average fitness error between the optimal fitness of the current environment and the best-of-generation fitness at each generation, is reported here since for DOPs a single, time-invariant optimal solution does not exist, the goal is not to find the optima but to track their progression through the space as closely as possible. The average fitness error at time t can be calculated as follows:

$$e^*(t) = \frac{1}{t} \sum_{i=1}^{t} |f_b^* - f_b(t)| \tag{7}$$

where f_b^* is the fitness of the optimum and $f_b(t)$ is the fitness of the best solution achieved at generation t. If several populations are used, $f_b(t)$ is the best solution over all populations at generation t.

5 Experimental Results

The preliminary experiments were first carried out on TMIPSO under the two triggered generators with different settings, where s is set to 1.0 and Δe is set to 100. The maximum generation is set to 1000, which equals 10 environmental changes. Each experimental result is averaged over 100 runs with the same set of different random seeds. The experimental results are shown in Fig. 1.

From Fig. 1, it can be seen that the approach of restricting the running average fitness (*averfit*, see the lower 3 curves) performs significantly better than the approach of restricting the maximum distance (*maxdist*, see the upper 3 curves) in the competition between the two different triggered generators. And the effectiveness of varying the threshold $b2$ settings in the *maxdist* scheme seems to be very small since the performance curves always wind together. That is, varying $b2$ does not affect the performance of algorithms much. However, the situation is different for the *averfit* scheme. It can be seen that the performance curves almost superpose together when $b1$ is small (i.e., 0.001 or 0.01), but the performance declines clearly when $b1$ becomes too large (i.e., 0.1). Therefore, $b1$ should not be set too large. For our following experiments, the *averfit* triggered generator will be used and the threshold $b1$ is always set to 0.001.

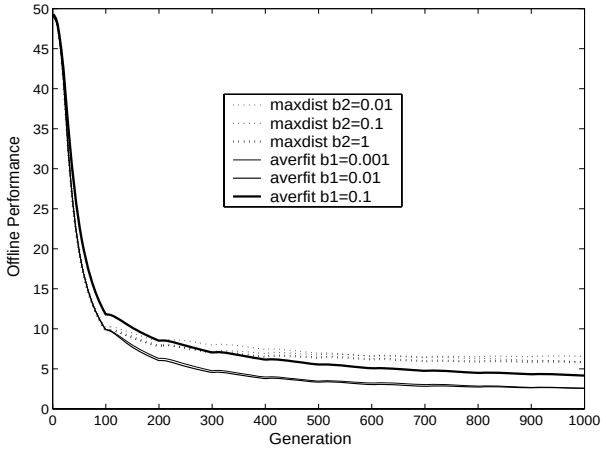
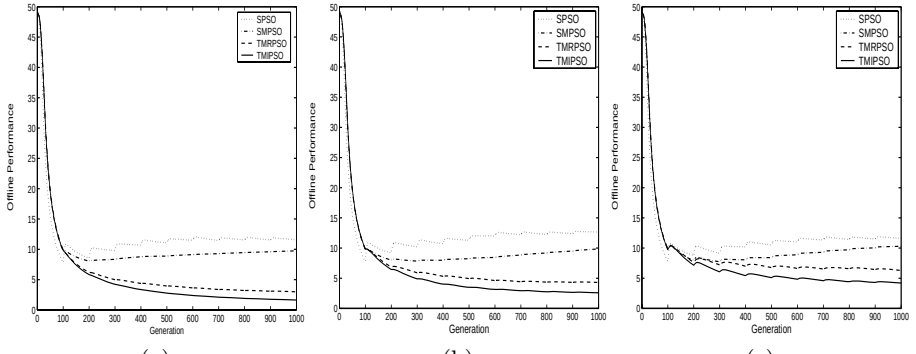


Fig. 1. Experimental results on the triggered memory-based immigrants PSO with different triggered generators in the dynamic environments



(a) (b) (c)

Fig. 2. Experimental results on the triggered memory-based immigrants PSO with different triggered generators in the dynamic environments with $\Delta e = 100$ and different severities of changes: (a) $s = 0.5$, (b) $s = 1.0$, and (c) $s = 2.0$

Fig. 2 plots the results of PSOs on the dynamic problems with different severities and Δe=100. From Fig. 2, several results can be observed.

First, SPSO slightly outperforms the memory-enhanced PSOs just for the stationary period (i.e., the first environment), but the memory-based PSOs always perform much better than SPSO for the dynamic periods. In the stationary period, all PSO models randomly search for the optimum in the solution space because the memory is empty and scanty. Hence, SPSO can more easily find a peak in the original fitness landscape since it has a single population whose size is much larger than each one of the populations in the memory-based PSOs.

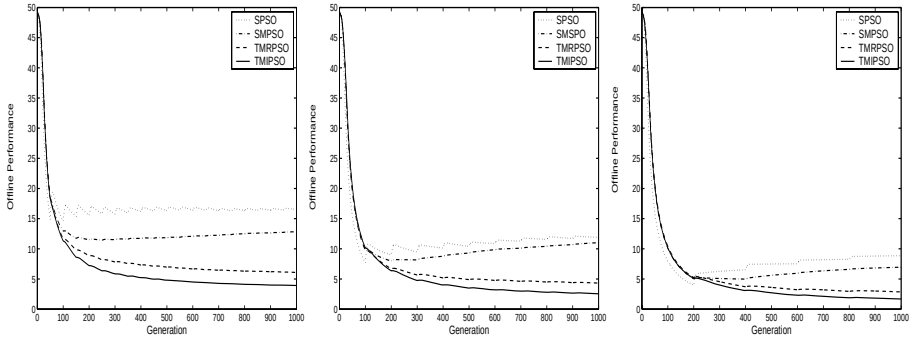


Fig. 3. Experimental results on four different PSO models in the dynamic environments with $s = 1.0$ and different frequencies of changes: (a) $\Delta e = 50$, (b) $\Delta e = 100$, and (c) $\Delta e = 200$

But in the dynamic periods, memory can help the population remember the past information and restart from the promising area closer to the new optimum. But SPSO will restart evolution from scratch here, which makes the population take a long time to reach an optimum (even a local optimum in most cases). This is also the reason why the performance curve of SPSO seems a little oscillatory.

Second, the triggered memory-based PSOs perform better than traditional memory-based PSO. In the triggered memory methods, the "explore"-population could contribute its best solution to the memory once a peak is affirmed to be found and the "exploit"-population is also injected with the new memory information at once. However, in the traditional memory method, the "explore"-population will contribute its solution and be re-initialized until a change is detected, which means that it could have stayed in a peak, if the peak is found, for a long time and lost the chance of finding a higher peak. Hence, the triggered memory methods can explore the solution space more efficiently and more quickly than the traditional memory method.

Third, the triggered memory-based resetting scheme performs worse than the triggered memory-based immigrants scheme over all the periods except at the beginning of evolution. In the memory-based resetting scheme, the information in the memory is reintroduced into the "exploit"-population just as an alternative of the global solution. This seems to just contribute an attractor to the population. However, whether the population could reach the neighbourhood of the attractor and keep the correct evolution in direction is not clear. Compared to the resetting scheme, the immigrants scheme employs a more efficient way where all the solutions in the memory are explicitly replaced into the "exploit"-population. On the other hand, the injection of more memorial information also helps the population maintain a high diversity in the immigrants scheme than in the resetting scheme.

Fourth, the change severity, which is one aspect of the environmental dynamism, affects the performance of all PSOs. And it seems natural that for

a fixed value of Δe, the performance of PSOs decreases when the value of s increases.

The experimental results on PSOs in the dynamic problems with different frequencies of environmental changes and $s = 1.0$ are plotted in Fig. 3. Similar results can be observed from Fig. 3 as from Fig. 2. The frequency of changes is another aspect of the environmental dynamism, and also naturally, when the frequency of change decreases, i.e., when Δe increases, the performance of all PSOs increases.

6 Conclusions

This paper investigates the application of PSOs with the tri-island memory model for DOPs. For this memory-based PSO model, a traditional way is to re-initialize the "explore"-population and retrieve the memory whenever an environmental change is detected. However, there may be some problems for this scheme. When the environment changes slowly, the population might have always stayed in a peak instead of searching for other peaks for a long time. In order to solve this problem, a new triggered memory scheme is proposed for the memory-based PSO in dynamic environments, where a triggered memory generator is designed for the retrieval period of memory. In this scheme, whenever the "explore"-population finds a peak, it will be immediately re-initialized and the memory will be retrieved. In order to determine that a peak has been found by the "explore"-population, two measures are also proposed. To retrieve the memory this paper proposes two strategies: the memory-based immigrants scheme replaces the solutions in the memory explicitly into the "exploit"-population and the memory-based resetting scheme only resets the record of the best global solution for the "exploit"-population using the best re-evaluated solution in the memory.

Based on the Moving Peaks Benchmark function [5], experiments were carried out to compare the performance of several PSOs including the proposed triggered memory-based PSOs in dynamic environments. From the experimental results, we can draw the following conclusions on the dynamic test problems. First, the memory mechanism can improve the performance of PSOs in dynamic environments. Second, the triggered memory method is more efficient than the traditional memory method in exploring the solution space. Hence, the triggered memory-based PSOs have stronger robustness and adaptability than the traditional memory-based PSO and simple PSO in dynamic environments, especially when the environment does not change frequently. Third, the memory-based immigrants scheme is more efficient than the memory-based resetting scheme for enhancing the performance of the triggered memory-based PSO in dynamic environments.

For future work, it is valuable to examine the performance of hybrid approaches that combine the triggered memory method and other approaches already known from the literature, e.g., the random immigrants scheme, for PSOs

in dynamic environments. In addition, it is also interesting to construct new triggered generators and examine them under the same framework.

References

1. T. Blackwell and J. Branke. Multi-swarm optimization in dynamic environments. In *Applications of Evolutionary Computing*, LNCS vol. 3005, pp. 489-500, 2004.
2. J. Branke. Memory enhanced evolutionary algorithms for changing optimization problems. In *Proc. of the 1999 IEEE Congress on Evolutionary Computation*, Washington, DC, USA, vol. 3, pp. 1875-1882, 1999, IEEE Press.
3. J. Branke, T. Kaußler, C. Schmidth, and H. Schmeck. A multi-population approach to dynamic optimization problems. *Proc. of the 5th Int. Conf. on Adaptive Computing in Design and Manufacturing*, pp. 299–308, 2000.
4. J. Branke. *Envolutionary Optimization in Dynamic Environments*. Kluwer Academic Publishers, 2002.
5. J. Branke. The moving peaks benchmark website. Online, http://www.aifb.uni-karlsruhe.de/jbr/MovPeaks.
6. A. Carlisle and G. Dozier. Adapting particle swarm optimization to dynamic environments. In *Proc. of the 2000 Int. Conf. on Artificial Intelligence*, Las Vegas, USA, pp. 429-434, 2000.
7. R. Eberhart and J. Kennedy. A new optimizer using particle swarm theory. In *Proceedings of the 6th International Symposium on Micro Machine and Human Science*, Nagoya, Japan, pp. 39-43, 1995, IEEE Press.
8. R. Eberhart and Y. Shi. Tracking and optimizing dynamic systems with particle swarms. In *Proc. of the 2001 IEEE Congress on Evolutionary Computation*, Seoul, Korea, vol. 1, pp. 94-100, 2001, IEEE Press.
9. D. E. Goldberg and R. E. Smith. Nonstationary function optimization using genetic algorithms with dominance and diploidy. In *Proc. of the 2nd Int. Conf. On Genetic Algorithms*, pp: 59-68, 1987.
10. J. J. Grefenstette. Genetic algorithms for changing environments. In R. Maenner and B. Manderick, editors, *Parallel Problem Solving From Nature 2*, pp. 137-144, 1992.
11. X. Hu and R. Eberhart. Adaptive particle swarm optimization: Detection and response to dynamic systems. In *Proc. of the 2002 IEEE Congress on Evolutionary Computation*, Hawaii, USA, pp. 1666-1670, 2002. IEEE Press.
12. J. Kennedy and R. Eberhart. Particle swarm optimization. In *Proc. of the 1995 IEEE Int. Conf. on Neural Networks*, Perth, Australia, vol. 5, pp. 1942-1948, 1995, IEEE Press.
13. R. W. Morrison and K. A. De Jong. Triggered hypermutation revisited. In *Proc. of the 2000 IEEE Congress on Evolutionary Computation*, San Diego, USA, pp. 1025-1032, 2000, IEEE Press.
14. D. Parrott and X. Li. A particle swarm model for tracking multiple peaks in a dynamic environment using speciation. In *Proc. of the 2004 IEEE Congress on Evolutionary Computation*, Portland, USA, pp. 98-103, 2004, IEEE Press.
15. S. Yang. Memory-based immigrants for genetic algorithms in dynamic environments. *Proc. of the 2005 Genetic and Evolutionary Computation Conference*, vol. 2, pp. 1115–1122, 2005.
16. S. Yang and X. Yao. Experimental study on population-based incremental learning algorithms for dynamic optimization problems. *Soft Computing*, **9**(11): 815-834, 2005.

Experimental Comparison of Replacement Strategies in Steady State Genetic Algorithms for the Dynamic MKP

A. Şima Uyar

Computer Engineering Department, Istanbul Technical University
Maslak 34469 Istanbul, Turkey
etaner@itu.edu.tr

Abstract. In the steady-state model for genetic algorithms (SSGA), the choice of a replacement strategy plays an important role in performance. Being able to handle changes is important for an optimization algorithm since many real-world problems are dynamic in nature. The main aim of this study is to experimentally compare different variations for basic replacement strategies in a dynamic environment. To cope with changes, a very simple mechanism of duplicate elimination is used. As an example of a dynamic problem, a dynamic version of the multi-dimensional knapsack problem is chosen. The results obtained here are in keeping with previous studies while some further interesting results are also obtained due to the special landscape features of the chosen problem.

1 Introduction

There are two major population models in genetic algorithms (GA): namely a generational model and a steady-state model. In the generational model (GGA), in each iteration, the number of offspring generated is equal to the population size and the offspring fully replace the old population. In the more common form of the steady-state model (SSGA), at each iteration, one offspring is generated, usually from two parents. A replacement strategy determines which individual in the current population is replaced by the newly created offspring. In [10], several replacement strategies for SSGAs in stationary environments are compared based on the expected take-over time of an individual in a finite population through a Markov chain analysis. Their results show that different replacement strategies have a great effect on the take-over times. The authors further extend their work with an experimental analysis of the replacement strategies in dynamic environments in [11]. In [8], several replacement schemes are compared based on the amount of diversity they preserve in the population and a new scheme is introduced which is based on the diversity contribution in addition to the fitness contribution of the individuals to the population. They mostly test their approaches in dynamic environments. The replacement strategies used in this study are mainly based on the ones discussed in [11] and [8].

Coping with changes for an optimization algorithm is important since many real-world problems are dynamic in nature. When working in dynamic environments,

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 647–656, 2007.
© Springer-Verlag Berlin Heidelberg 2007

tracking the optima as well as quick adaptation to the change becomes important. There are many studies dealing with techniques proposed for GAs in dynamic environments. Detailed information can be obtained from [2]. The main aim of this study is to compare the performances of different replacement strategies for a dynamic environment. Comparing different techniques proposed for dynamic environments is out of the scope of this paper. In [3], it is noted that a simple duplicate elimination provides the sufficient diversity to cope with changes, so only duplicate elimination is used in this study. The results obtained here are in keeping with previous studies while due to the special landscape features of the dMKP, some interesting results are also obtained. This work reports the results of a preliminary study. The interesting results obtained here promote further study.

The paper is organized as follows: Section 2 explains the dMKP problem and its implementation. Section 3 introduces the replacement strategies for SSGAs used in this study. In Section 4, the experimental design and the results of the experiments are given along with a discussion of these results. Section 5 concludes the paper and provides possible future study options.

2 Dynamic Multi-dimensional Knapsack Problem

The multi-dimensional knapsack problem (MKP) is an NP-complete combinatorial optimization problem which has a wide range of real world applications such as cargo loading and budget management. The objective function for the MKP is given in Eq. 1.

$$\begin{aligned} &maximize \quad \sum_{j=1}^{n}(x_j \,.\, p_j) \\ &subject\ to \quad \sum_{j=1}^{n}(x_j \,.\, r_{ij}) \leq C_i \qquad i = 1, 2, ..., m \end{aligned} \tag{1}$$

where n is the number of items, m is the number of resources, $x_j \in \{0, 1\}$ shows whether item j is included in the subset or not, p_j shows the profit of item j, r_{ij} shows the resource consumption of item j for resource i and C_i is the capacity constraint of resource i.

In this study, a penalty approach for infeasible individuals is used. Each location on the chromosome corresponds to an item and shows whether the item is included in the solution or not. The penalty calculation method recommended in [7] is employed. The fitness of an individual is calculated based on its objective value calculated using Eq. 1 minus the penalty term. The dMKP implementation proposed in [4] is implemented. A change means that the profits, the resource consumptions and the constraints are multiplied by a normally distributed random variable as shown in Eq. 2:

$$\begin{aligned} p_j &\leftarrow p_j * (1 + N(0, \sigma_p)) \\ r_{ij} &\leftarrow r_{ij} * (1 + N(0, \sigma_r)) \\ c_i &\leftarrow c_i * (1 + N(0, \sigma_c)) \end{aligned} \tag{2}$$

Each profit p_j, resource consumption r_{ij} and constraint c_i is restricted to an interval as determined in Eq. 3.

$$\begin{gathered} lb_p * p_j \leq p_j \leq ub_p * p_j \\ lb_r * r_{ij} \leq r_{ij} \leq ub_p * r_{ij} \\ lb_c * c_i \leq c_i \leq ub_p * c_i \end{gathered} \tag{3}$$

where lb_p, lb_r, and lb_c are the multipliers to obtain the lower bounds and ub_p, ub_r, and ub_c are the multipliers to obtain the upper bounds. If any of the changes causes any of the bounds to be exceeded, the value is set to the boundaries.

3 Replacement Strategies in SSGA

In the most common form of the SSGA, at each iteration, one offspring is generated, usually from two parents through crossover and mutation. Further details on SSGAs can be found in [6]. The choice of a replacement strategy plays an important role in the performance of the SSGA. The main groups of the strategies used here are: replace a randomly selected individual (RR), replace the oldest individual (RO), replace the most similar individual (RMS) and replace the worst individual (RW). In RR, the new individual replaces a randomly selected individual in the current population. In RO, the oldest individual in the population is chosen for replacement. If there are more than one individuals with the same age, the one with the lowest fitness value is selected. In RMS, the new individual replaces the most similar individual in the population. Since a binary representation for chromosomes is used in this study, the similarity between two individuals is determined by the Hamming distance between them. The individual which has the smallest Hamming distance to the offspring is replaced. If there is more than one such individual, the one with the lowest fitness value is chosen for replacement. In RW, the individual with the lowest fitness value is replaced.

For the experiments in this study, unless otherwise stated, for each one of these strategies five variations are explored. In variations 1, 2, 4, a replacement always occurs. In variations 3, 5, replacement only occurs if the new individual has a better fitness than the individual chosen for replacement.

In variation 1, the pure forms of each of the strategies is used. Here the individuals to be replaced are selected from the whole population. In variation 2 (E), elitism is introduced by excluding the current best individual from the replacement candidates. This method ensures that the best individuals are not replaced. In variation 3 (B), the individuals to be replaced are determined from the whole population; however, replacements only take place if the fitness of the new individuals are better than the individuals chosen to be replaced. These variations inherently have elitism and thus ensure that the best individuals are not replaced. In variation 4 (KT), a random individual is selected in addition to the individual chosen for replacement. Whichever of these has a lower fitness, gets replaced. These variations also have inherent elitism. Variation 5 (KTB) is a modification of the previous variation. Here, a new individual replaces a member

of the population only if it has a better fitness than the individual initially chosen for replacement and the random individual. RKTB also has inherent elitism.

4 Experimental Design

The aim of the experiments is to explore and explain the performance and behavior of the different replacement strategies for the dMKP. The MKP instance used as the base problem is selected from the OR-LIBRARY [1][1]. It has 100 items, 10 knapsacks and a minimum tightness ratio of 0.75. The minimum tightness ratio is related to the hardness of a problem. A higher value means an easier problem, so the chosen base instance is relatively easy. The standard deviations of the normally distributed random variables used in Eq. 2 is set to $\sigma_p = \sigma_r = \sigma_c = 0.05$. All the lower bound multipliers in Eq. 3 are set to 0.5 and the upper bound multipliers are set to 1.5. These values determine the severity of the change and the settings used in this study denote a moderate change severity.

A modified version of the offline performance is used for comparisons. The offline performance for dynamic environments [2] gives a running average of the best individuals found. However, for the dMKP, in some cases, many of the individuals are infeasible. The penalty mechanism assigns very large negative fitness values to infeasible individuals, thus averaging these with the few feasibles gives meaningless results. Because of this, instead of averages, for each fitness evaluation, the best fitness found so far is taken (The first fitness evaluation for each new environment is automatically taken similar to the calculation of offline performance.). The performance comparisons are made through plots of the chosen metric. In all the plots, the x-axis shows the fitness evaluation count and the y-axis shows the best fitness found so far. The y-axis is started from '0.0', which means that some of the strategies which are not able to produce feasible results are not seen on the plots. This is done in order to make the plots more readable. However, discussions are given based on the complete plots not shown here. In all further results and discussions, the strategies are abbreviated as given in Table 1.

For all tests, the SSGA given in Section 3 is used. Each time an environment change occurs, the existing individuals are not re-evaluated. In this case, since changes are not needed to be detected, it also removes the possibility of the harmful effects of false detections [9]. Parents are selected randomly because individuals are not re-evaluated after a change. No extra approaches for dynamic environments, such as hypermutation [5], is used. In [3], it is noted that a simple duplicate elimination is sufficient to handle the dynamics introduced by the dMKP used in this study. A population consists of 50 individuals; uniform crossover is performed with probability $p_{cr} = 1$ and point-mutation is applied with probability $p_m = 1/L$ where L is the chromosome length which is 100. The SSGA is run for a total of 21000 fitness evaluations with 1000 fitness evaluations between each environment change. This gives a total of 20 environment changes.

[1] The 21st problem in the "mkpnapcb4.txt" is used.

Table 1. Abbreviations of Replacement Strategies used in Experiments

RR	Replace random
RRB	Replace random if better
RRE	Replace random with elitism
RRKT	Replace after tournament between two random individuals
RRKTB	Replace after tournament between two random individuals if better
RO	Replace oldest
ROB	Replace oldest if better
ROE	Replace oldest with elitism
ROKT	Replace after tournament between oldest and random individual
ROKTB	Replace after tournament between oldest and random individual if better
RMS	Replace most similar
RMSB	Replace most similar if better
RMSE	Replace most similar with elitism
RMSKT	Replace after tournament between most similar and random individual
RMSKTB	Replace after tournament between most similar and random individual if better
RW	Replace worst
RWB	Replace worst if better

The chosen values correspond to 20 generations between each change and can be considered a moderate frequency of change. All results are given averaged over 20 runs of the corresponding programs.

The results for the RR group of strategies are given in Fig. 1. RRKT is the best performer, while RR is the worst. RRB and RRKTB start out good but start deteriorating with the changes. Since these variations tend to keep the individuals with better fitnesses (though they may have been evaluated in previous environments), after a few changes the population gets filled with previous good individuals who may not be good for the current environment. This also explains why RRKT outperforms RRKTB. RRE starts worse than RRB and RRKTB, keeps improving for a few environments and again starts deteriorating after several changes. A pure random replacement strategy performs the worst as expected since it has no mechanism for preserving good individuals. However, RRB and RRKTB perform better than RRE because the last one is a pure elitist strategy. Even though RRB and RRKTB try to preserve better individuals, they do not explicitly try to preserve the best. In the SSGA in this study, individuals are not re-evaluated after a change, thus the elite individual is likely to be an individual evaluated using previous environments. For the RRB ad RRKTB, some of the better ones kept may be those evaluated in the current environment.

The results for the RO group of strategies are given in Fig. 2. ROB is the best though it degrades towards the end. ROKTB starts out similar to ROB, but starts degrading earlier. ROKT starts out bad, improves along the changes but cannot reach the performance of the previous two. RO starts out bad, gets worse and finishes as the worst. ROE starts out even worse than RO, improves along the changes and finishes better than RO but worse than the others. This strategy

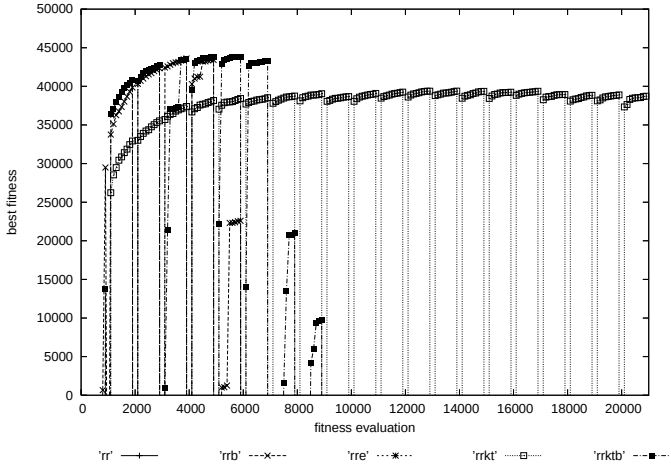


Fig. 1. Plots for RR group of strategies

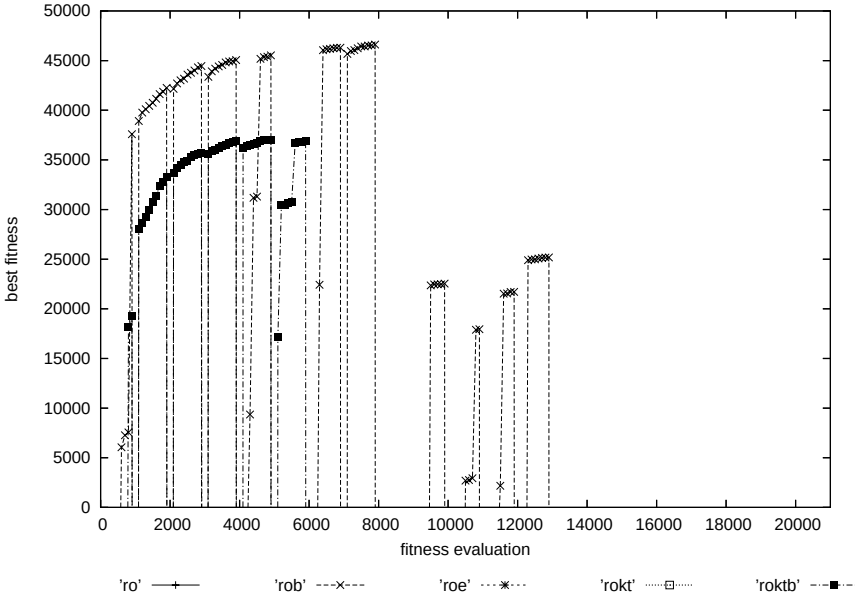


Fig. 2. Plots for RO group of strategies

uses an aging mechanism where old individuals are removed from the population. This implies that individuals evaluated in previous environments get replaced, so using a mechanism which compares two individuals based on their fitnesses is more successful than a pure strategy since it is likely to compare individuals evaluated for the same environment while preserving good individuals to some

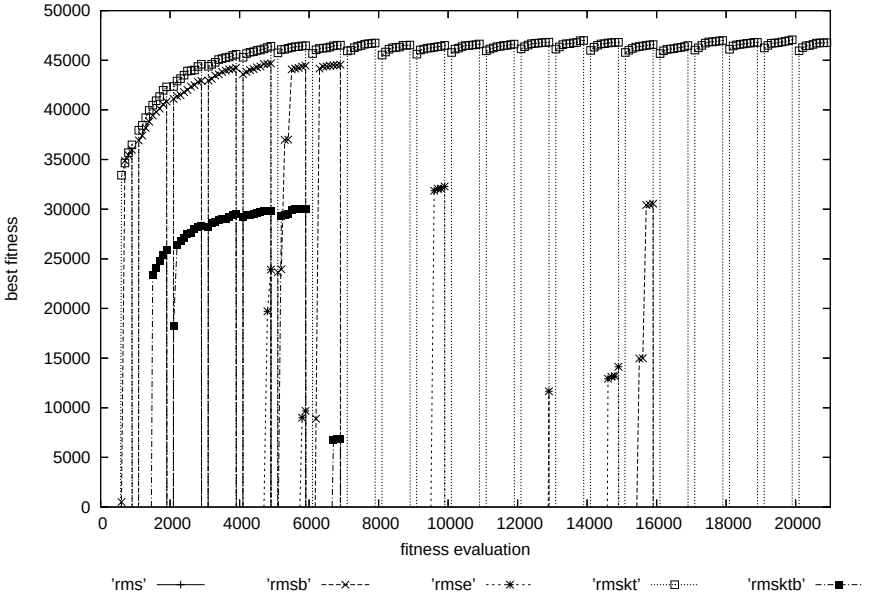


Fig. 3. Plots for RMS group of strategies

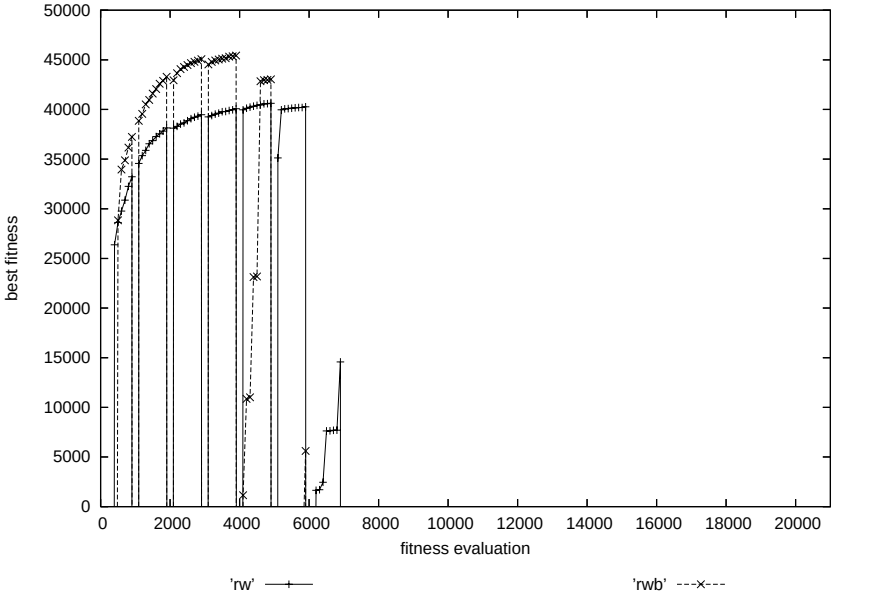


Fig. 4. Plots for RW group of strategies

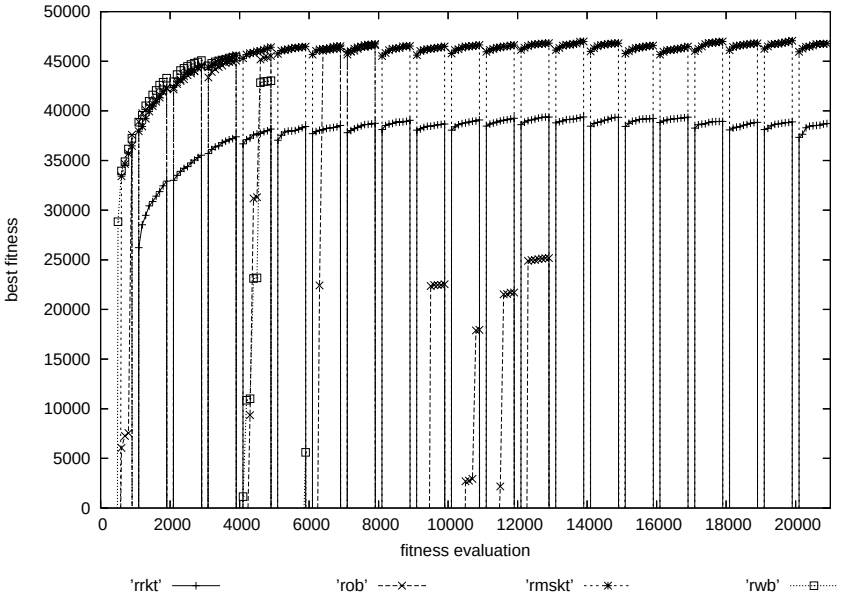


Fig. 5. Plots for best strategies from each group

extent. However, an elitist technique is not successful because it preserves the best regardless of its age. The elite individual is likely to be one evaluated in previous environments.

The results for the RMS group of strategies are given in Fig. 3. As can be seen, RMSKT is the best though it improves slower than RMSB. However, even though RMSB starts out well, it deteriorates with the changes. RMSE and RMSKTB perform quite similarly to RMSB. As is the case with the previous strategies, the pure strategy is the worst performer. The ordering and general behavior of the RMS group of strategies resembles those of the RR group, but they are slower. In [7] it is shown that the penalty function, which is also used in this study, drives the population to the boundary of feasibility. When the population reaches the boundary, the individuals become close (similar) to each other. So replacing the most similar individual in the population becomes equivalent to replacing a random individual. The reason why the RMS group of strategies are slower than the RR group can be explained by the fact that the similarity to the random replacement effect occurs after the population is located along the boundary of feasibility which takes some time.

The results for the RW group of strategies are given in Fig. 4. Since these strategies rely heavily on the relative fitness of individuals and are inherently elitist, they both perform badly and are not able to cope with the changes.

Since changes occur every 1000 fitness evaluations, the first environment can be regarded as a stationary MKP. The performance comparisons of these are discussed in [10]. The reason why some strategies start out bad is based on their

performances in the stationary case. The results obtained here are in keeping with those in that study.

The best performer from each group of strategies is plotted in Fig. 5. RRKT and RMSKT have a similar performance at the end, but RMSKT takes longer to achieve this than RRKT. In addition to this, in RMSKT, for each replacement decision, the Hamming distance of the new individual to all the individuals in the current population has to be computed. This causes RMSKT to be computationally more expensive than the RKT. ROB and RWB start out similar to RMSKT but RWB starts deteriorating very early on in the iterations while ROB deteriorates towards the end. The reason for the similar performances of RMSKT and RRKT are explained above. Even though ROB is the best performer among the RO group of strategies, it is not better than RMSKT and RRKT, both of which do not use the fitness information in the same way ROB does. The aging mechanism is able to help but still it suffers from the fact that individuals evaluated in previous environments exist together with those evaluated in the current environment. In both ROB and RWB, a replacement does not always occur so the chances of the individuals evaluated in the current environment filling up the current population is lower than RRKT and RMSKT where a replacement always occurs. The effect of the aging mechanism explains the performance difference between ROB and RWB.

These results obtained in this study are mainly in keeping with those obtained in [11]. In that study, the RW strategy is the overall worst performer and in the cases where there is no re-evaluation, the elitist strategies perform very poorly. It is noted that strategies which involve a conservative approach (i.e. similar to RRB, ROB, RMSB, RWB), improve the performance of their pure forms. However, even though elitism is not directly enforced, the best individual is preserved if it is selected for replacement, so they perform better than those using a direct elitism. The RMS group of strategies are not discussed in [11] but they are included in this study since they have a special behavior pattern for the MKP problem due to the penalty function used.

5 Conclusion and Future Work

It is known that replacement strategies play an important role on the performance of the GA both in stationary and dynamic environments. The results obtained in this study are in keeping with those of previous studies, however some interesting behaviors of the strategies due to the properties of the MKP landscape are seen here. The main issue in this study is the fact that individuals are not re-evaluated after a change. Assuming that the most costly operation is the fitness calculations, this approach saves on the number of fitness evaluations. It also has another advantage that it does not require the changes to be made explicitly known to the system since a faulty change detection is detrimental to the performance of a GA [9]. However, in [11], it is strongly stressed that re-evaluation is useful. Based on this, further experiments with the dMKP using the strategies in this study should be performed as a future study. Another

future study is to look at the performance of these strategies in other types of dynamic environments. The MKP is a constrained problem and thus has feasible and infeasible solutions. Using a penalty approach as the one in this study, introduces very deep holes in the landscape. The results obtained in this study may be specific to problems having such landscapes. In order to be able to generalize the results, experiments with other types of landscapes are needed. Overall, this study provides interesting results and thus it promotes further study.

References

1. J. E. Beasley. Or library. *http://www.brunel.ac.uk/depts/ma/research/jeb/info.html*, 2004.
2. J. Branke. *Evolutionary Optimization in Dynamic Environments*. Kluwer Academic Publishers, 1st edition, 2002.
3. J. Branke, M. Orbayi, and S. Uyar. The role of representations in dynamic knapsack problems. In *3rd European Workshop on Evolutionary Algorithms in Stocahstic and Dynamic Environments*, pages 764–775. Springer, 2006.
4. J. Branke, E. Salihoglu, and S. Uyar. Towards an analysis of dynamic environments. In *Genetic and Evolutionary Computation Conference*, pages 1433–1439. ACM, 2005.
5. H. G. Cobb. An investigation into the use of hypermutation as an adaptive operator in genetic algorithms having continuous, time-dependent nonstationary environments. Technical Report AIC-90-001, Naval Research Laboratory, Washington, USA, 1990.
6. A. E. Eiben and J. E. Smith. *Introduction to Evolutionary Computing*. Springer, 1st edition, Nov. 2003.
7. J. Gottlieb. On the feasibility problem of penalty-based evolutionary algorithms for knapsack problems. In *Proceedings for EvoWorkshops: EvoCOP*, 2001.
8. M. Lozano, F. Herrera, and J. R. Cano. Replacement strategies to maintain useful diversity in steady-state genetic algorithms. In *Proceedings of the 8th Online World Conference on Soft Computing in Industrial Applications*, Sept. 2003.
9. R. W. Morrison. *Designing Evolutionary Algorithms for Dynamic Environments*. Springer, 1st edition, 2004.
10. J. Smith and F. Vavak. Replacement strategies in steady state genetic algorithms: Static environments. In *Proceedings of Foundations of Genetic Algorithms V*, pages 219–233. Morgan Kaufmann, 1998.
11. J. E. Smith and F. Vavak. Replacement strategies in steady state genetic algorithms: Dynamic environments. *Journal of Computing and Information Technology, Special Issue on Evolutionary Computing*, 7(1), Mar. 1999.

Understanding the Semantics of the Genetic Algorithm in Dynamic Environments

A Case Study Using the Shaky Ladder Hyperplane-Defined Functions

Abir Alharbi[1], William Rand[2], and Rick Riolo[3]

[1] King Saud University, Mathematics Department, Riyadh, 11495, Saudi Arabia
a_alharbi@mail.com
[2] Northwestern University, Northwestern Institute on Complex Systems, Evanston, IL, 60208-4057, USA
wrand@northwestern.edu
[3] University of Michigan, Center for the Study of Complex Systems, Ann Arbor, MI 48109-1120, USA
rlriolo@umich.edu

Abstract. Researchers examining genetic algorithms (GAs) in applied settings rarely have access to anything other than fitness values of the best individuals to observe the behavior of the GA. In particular, researchers do not know what schemata are present in the population. Even when researchers look beyond best fitness values, they concentrate on either performance related measures like average fitness and robustness, or low-level descriptions like bit-level diversity measures. To understand the behavior of the GA on dynamic problems, it would be useful to track what is occurring on the "semantic" level of schemata. Thus in this paper we examine the evolving "content" in terms of schemata, as the GA solves dynamic problems. This allows us to better understand the behavior of the GA in dynamic environments. We finish by summarizing this knowledge and speculate about future work to address some of the new problems that we discovered during these experiments.

1 Introduction

It has been speculated that the genetic algorithm (GA) makes use of higher levels of content like building blocks and their formalized descriptions in the form of schemata [1]. However, this hypothesis has been hard to substantiate because researchers only observe performance-related measures, like fitness of the best individuals, and low-level descriptions of the GA population, like bit-wise diversity measures. In order to understand the GA in more depth researchers have created the hyperplane-defined functions (hdfs) [2] and the shaky ladder hyperplane-defined functions (sl-hdfs), for dynamic environments [3]. The hdfs (and the sl-hdfs) are designed to represent the way the GA searches by combining building blocks, hence they are appropriate for examining the behavior

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 657–667, 2007.
© Springer-Verlag Berlin Heidelberg 2007

of the GA at a higher level than just that of bitwise operations and performance. Moreover, the sl-hdfs reflect a large class of problems where there is a global unchanging optimum which has regularly occurring subproblems, but the rewards for those subproblems change in time [4]. Thus the sl-hdfs are a good test suite for exploring and understanding the use of the GA in dynamic environments. However, the sl-hdfs are not designed to provide a benchmark of how well different GA variants perform, nor are they meant to classify the underlying environments. Instead the intent of the sl-hdfs is to understand the behavior of the GA operating within dynamic environments.

Earlier work with the sl-hdfs has resulted in observations about the GA that were not adequately explained by performance-based measures. For instance, the GA performs better in dynamic environments than in static environments [3]. Second, the GA performs better in dynamic environments where transitions, i.e. changes in the underlying fitness function, are rugged as opposed to those where the environmental changes are smooth [5]. We advance one hypothesis to explain both of these observations. In both the static case and the smooth transition case, the GA becomes stuck on local optima. However, using standard methods this hypothesis is difficult to confirm. Thus, in order to substantiate this hypothesis we decided to look beyond examining just bit evolution or the "syntax" of the GA. Instead, we decided to examine higher level components of individuals in the GA (in this case schemata) and their composition within the population; this allows us to view the "semantics" of the GA. This was in part inspired by similar work in schemata analysis within genetic programming and static environments [6].

We begin this paper by recapping previous observations on performance metrics. Then we describe new schemata measures that we developed, and present the results of our semantic experiments. We discuss these results, and the new questions that they raise, before concluding and discussing future work.

2 Previous Mysteries of the sl-hdfs

For these experiments we utilize the sl-hdfs [3], which are a restricted form of Holland's hdf [2]. These restrictions guarantee that any string that matches the highest level schema must be optimally valued. Moreover they give us an easy way to create a similar but different sl-hdf by changing the intermediate building blocks. A more in-depth explanation of the construction of the sl-hdfs has been presented in previous work [4].

There are many parameters that control the construction of the sl-hdfs, and we group these settings into variants [5]. For the purpose of this paper, it is mainly necessary to know how the variants affect the transitions in the fitness function; that is how the fitness landscape changes when the underlying function is modified. All sl-hdfs are composed of elementary schemata, intermediate schemata, potholes, and a highest level schema. Transitions, called "shakes of the ladder", occur when the intermediate schemata are altered. Since the intermediate schemata are the only things that are altered the differences between the

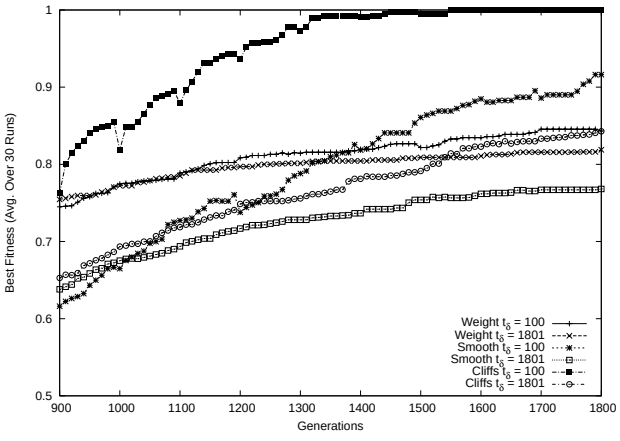


Fig. 1. Fitness for all three variants with $t_\delta = 100$ and 1801

variants of the sl-hdfs revolve around how intermediate schemata change. The "Cliffs" variant has the sharpest transitions, which means that the landscape looks very different after the "shake." The "Weight" variant has the smoothest transitions and the "Smooth" variant is somewhere in between, for other parameter differences see Table 1. Besides the variant being explored, there is one other variable of interest that will be manipulated, which is t_δ. t_δ controls the number of generations between a change in the sl-hdf. $t_\delta = 1801$ represents a static environment, since the runs we will be presenting are only observed for 1800 generations, and $t_\delta = 100$ represents a dynamic environment. On the basis of previous results, we have found that $t_\delta = 100$ provides a good setting for understanding how the GA behaves in regularly changing environments [4].

In Figure 1, we recapitulate the results from two previous papers [5] [3], that illustrate the two observations mentioned in the introduction: (1) In all cases (the three variants) at the end of the runs, the GA operating in the dynamic environment does at least as well as, and in most cases does better than, it does in the static environment. (2) In all cases (static vs. dynamic) at the end of the runs, the GA performs better in the Cliffs variant than in the other variants.

3 The Experiments and Results

The basic setup for our experiments is a simple GA using the sl-hdf as a fitness function. The GA uses one-point crossover, per bit mutation, and full population replacement. In these experiments, we set the number of generations between shakes of the ladder (t_δ) to 100, which represents a moderate rate of change. The optimal value is 1.0. All results below are presented at 10 generation increments to make the graphs easier to read. Detailed parameter settings are in Table 1.

Table 1. Basic GA and sl-hdf Parameters

Parameter	Cliffs Variant	Smooth Variant	Weight Variant
Population Size	1000		
Mutation Rate	0.001		
Crossover Rate	0.7		
Generations	1800		
String Length	500		
Selection Type	Tournament, size 3		
Number of Elem. Schemata	50		
Elementary Schemata Order	8		
Elementary Schemata Length	Not Specified		50
Mean, Var. of Int. Schem. Wt.	3, 0		3, 1
Int. Constr. Method	Unrestr., Random	Restr., Random	Restr., Neighbor
t_δ	100		
w_δ	0		1
Number of Runs	30		

4 Average Schemata Analysis

One of the principle arguments as to why the sl-hdfs are a good test bed for exploring the GA in dynamic environments is that the sl-hdfs are built in the same way that the GA works. In other words, the sl-hdfs are built out of elementary schemata which are combined to create longer schemata. The Building Block hypothesis [1] states that the GA works by combining schemata, which are partial solutions to a problem, in order to create more complicated, but better solutions. Thus in order to understand the behavior of the GA in dynamic environments, it helps to understand how the schemata in the population evolve over time.

In our experiments, the fraction of the schemata, on average, present in the population is calculated for each generation. We examine each individual in the population and we look at how many schemata of a particular set (elementary, potholes, highest-level, intermediate) that individual has identified and divide by the total number of schemata it is possible to identify in that set. We then sum this number up across the whole population for that generation. We then divide by the number of the individuals in the population (1000) to get the fraction of that set of schemata that each individual on average in the population has discovered. Finally, we average this result at the current generation across all runs (30). The average fraction of all 50 elementary schemata each generation is determined along with the average fractions of all 47 intermediate schemata, all 50 pothole schemata and the highest level schema.

4.1 Cliffs Variant Schemata Analysis

We begin by analyzing the Cliffs variant. The average schemata results are plotted in Figure 2. The intermediate schemata fraction shows the most change: after every shake of the ladder it drops. This is because the set of intermediate schemata rewarded by the sl-hdf changes at each shake. Thus the population has a smaller fraction of the intermediate schemata that now are rewarded by

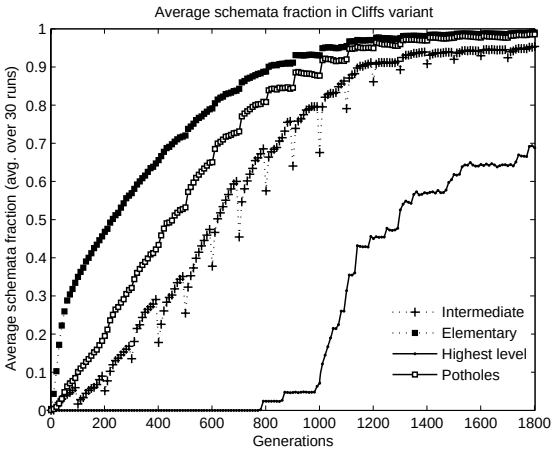


Fig. 2. Cliffs Variant: Schemata Analysis with $t_\delta = 100$

the sl-hdf compared to the fraction it had found before the shake. However, the population immediately starts to acquire the newly rewarded intermediate schemata, and thus the fraction of intermediate schemata quickly surpasses the old values. Since many intermediate schemata contain potholes, the population acquires new potholes as well, which is why the fraction of pothole schemata also increases sharply. As can be seen from Figure 2, some of the runs start to find the highest level schema as early as generation 800, and by generation 1800, 70% of the runs have found it. The severe changes of this variant prevents the population from being locked into particular building blocks and forces it to explore a wider range of schemata.

4.2 Smooth Variant Schemata Analysis

In Figure 3, we present the results for the Smooth variant. The intermediate schemata show small drops after each shake, since there is not as much difference between the Smooth variant's potential intermediate schemata, and thus the shakes in this variant are more smooth than in the Cliffs variant. The fraction of intermediate schemata also rebounds after the shake to surpass the fraction before the shake like in the Cliffs variant, but since the Smooth variant does not perturb the population as much as the Cliffs variant, the increase is not as sharp. Again, the fraction of potholes also increases due to intermediate schemata that contain potholes. Moreover, the highest level schema fraction is zero for most generations, because few of the runs find an optimal string until near the end. In general the graphs in this variant are smoother than they are in the Cliffs variant, since the shakes are smoother. Despite the smooth transitions, the Smooth variant does not perform as well as the Cliffs variant. The differences between Figure 2 and Figure 3 could be explained by the hypothesis that the

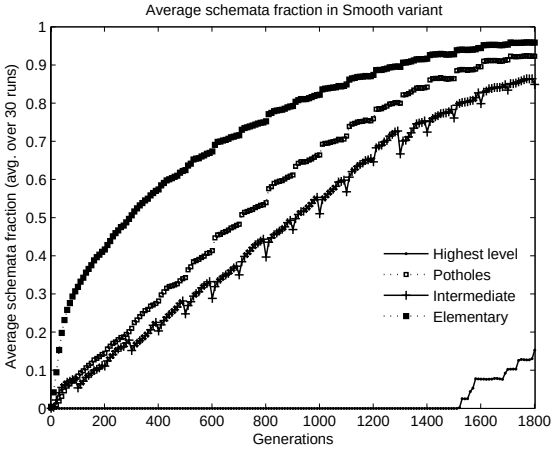


Fig. 3. Smooth Variant: Schemata Analysis with $t_\delta = 100$

unstable landscape in the Cliffs variant perturbs the population off local optima and thus it quickly rebounds from the lost schemata, and in fact winds up finding more intermediate schemata than it had before the shake. In the Smooth variant on the other hand, the population is not perturbed far enough and though the GA population finds the new intermediate schemata that are favored by the current ladder, these schemata are still within the basin of attraction of the previous local optima.

4.3 Weight Variant Schemata Analysis

In Figure 4, we present the results for the Weight variant. These plots are even smoother, but still this variant reaches lower fractions in the end than all other variants for all sets of schemata. Though the intermediate schemata has higher fractions on average than in the other variants for most of the run because they do not change in structure just in weights in this variant. One interesting result is that there is almost no effect on any of the results in the Weight variant due to the shakes of the ladder. Thus there is no gain from making the environment dynamic, like there is in the Cliffs and Smooth variants. In fact, this graph is similar to what you would expect for a GA operating on a static version of the hdf. In the static version, since there are no changes in the schemata and if we assume minimal disruption due to mutation and crossover, then all the lines should be monotonic as the GA accumulates first one schemata then another. The graph for the Weight variant is very similar to this expected graph for a static environment. This contrasts with previous results, where we compared the Cliffs variant with $t_\delta = 100$ and $t_\delta = 1801$ (a static environment); during that comparison we found that the dynamics in the environment helped the GA perform better. It seems to be the case that the transitions in the Weight variant are too smooth to affect the behavior of the GA [5]. In the end the GA operating

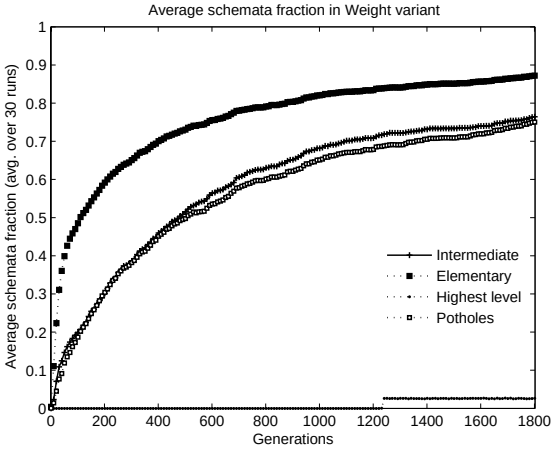


Fig. 4. Weight Variant: Schemata Analysis with $t_\delta = 100$

in the Weight variant underperforms the GAs operating in the Smooth and Cliffs variants, even though the GA was able to make much more rapid progress early on due to short elementary schemata.

4.4 Comparison of Results

The three experiments presented here (Cliffs, Smooth and Weight variants) show that schemata analysis allows to examine the population as it evolves on the fitness landscape in a different way. This analysis *explains* what is happening within the performance measures, which is after all a combination of what is happening at the schemata levels. Instead of just giving us raw performance measures for the GA, the schemata analysis tells us how the semantics of the fitness function, i.e. the schemata, are being assembled within the GA population.

For instance, the elementary schemata are contained by many more individuals than the other levels of schemata because they make up all other higher level intermediate schemata, therefore they have highest fractions, followed by the potholes. Since in the sl-hdfs potholes are a combination of other schemata, they are always accumulated at a rate less than that of the elementary schemata, but slightly faster or at the same rate as the intermediate schemata. One interesting difference is that the Cliffs variant seems to have a sharper increase in the fraction of potholes immediately after a shake when compared to the other two variants. This is because the nature of the unrestricted construction routine in the Cliffs variant [5] means that each shake of the ladder can "uncover" potholes that were canceled out by intermediate schemata in the structure of the sl-hdf before the shake. Moreover, the Cliffs variant is the only variant where the fraction of potholes is not monotonically increasing. Toward the end of the run it actually decreases at times indicating that the population is losing potholes that

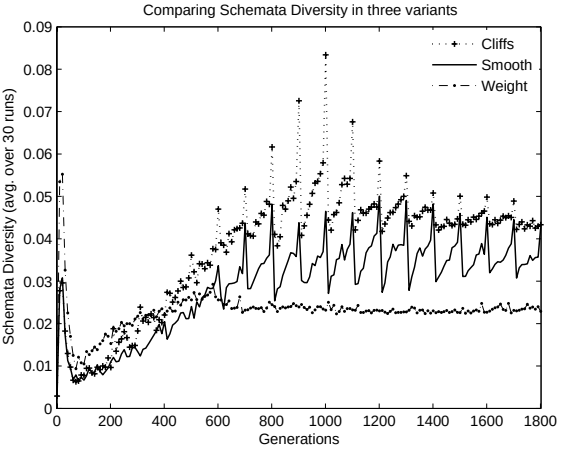


Fig. 5. Comparison of the Diversity of Schemata with $t_\delta = 100$

it had before. It is impossible not to acquire all of the potholes and acquire the highest level schema, since potholes are a subset of the highest level schema, thus this must be an intermediate effect, where certain areas of the search space are avoided by the population due to their negative fitness contribution. However, this same avoidance is not seen in the other two variants.

The intermediate schemata show the most change in their measures; except in the Weight variant, after every shake the fraction of individuals containing intermediate schemata decreases but it quickly returns and surpasses the value it had before the shake. Comparing the intermediate fractions in the different variants shows the highest values are in the Cliffs variant. In this variant they start rapidly increasing early on (as they do in the Weight variant) and continue to increase to reach the highest value while the Weight variant prematurely levels off. The short schemata length of the elementary schemata in the Weight variant allow it to make significant progress early on, but as can be seen the Weight variant prematurely levels off, indicating that it is no longer able to find new intermediate schemata. Thus it is the shaking of the ladder by form instead of Weight that prevents premature convergence in the sl-hdfs. This hypothesis is further substantiated by the fact that the Cliffs variant eventually outperforms the Weight variant, despite having elementary schemata of undefined lengths, and starting out slower.

5 Diversity of Schemata Analysis

Our examination of the diversity of the schemata in the populations measures how different the schemata present in each individual is as compared to the schemata found in all other individuals. Of course, diversity has been measured before at

the bit level [7]. In fact many GAs that are modified to work in dynamic environments specifically attempt to maintain diversity in the population [8]. However, examining schemata diversity provides a different look into the behavior of the GA than bit level diversity.

To measure schemata diversity we remap the individuals in the population from a bit-string s to a schemata-string s'. Individual x has a 1 at the ith location of s' if and only if s contains schema i. We then calculate the Hamming distance between individual x in s' and all other individuals in the population at generation t, and average the results to generate the average Hamming distance between that individual and all other individuals. We then average this result across all individuals. Finally we average this result across 30 runs.

As can be seen in Figure 5, the Cliffs variant undergoes two phases in schemata diversity. In the first phase schemata diversity increases dramatically as the GA explores the space of schemata. This phase lasts until about generation 1000 when many of the runs start to find optimally valued strings. In this second phase the shakes of the ladder have less of a dramatic effect on the schemata space since if an individual contains all the schemata then the shake will not affect it, or put another way: the shake only affects those that have not found the max solution. As the population converges on the equivalence class of optimal strings, the individuals in the population *have to be similar* in the semantic space. This is because in the end there is one optimally valued schema, and it contains all of the other schemata within it. Any individual that is optimally valued has a diversity measure of 0.0 from any other optimally valued individual; this is true despite the fact that the actual strings might be quite different, since the optimal schemata does not specify every bit. Thus, diversity in bit space (syntactic) increases as the population approaches the optimal in the sl-hdf because there are many neutral bits (see [9]); however, the diversity in the schemata space (semantic) decreases as the population approaches the optimal.

In the Smooth variant the same results can be seen but the magnitude of the changes is not as great, which supports the hypothesis that there is not as much selection pressure on the GA to explore beyond the local optima and thus all individuals quickly wind up looking similar.

Finally, in the Weight variant the schematic diversity is almost never affected by any of the changes in the environment, and it quickly reaches a plateau where it levels off. This indicates that the amount of exploration done by the GA is minimal, and that the populations look very homogeneous (within each run and at each generation) from generation 600 on (on average). The lack of any diversity and the fact that many intermediate schemata are never found indicate that the GA only explores a small portion of the Weight variant landscape.

6 Conclusion and Future Work

This paper analyzed the evolution of schemata in three different related dynamic environments. In particular, in this work we were able to take advantage

of our analysis of the schemata present in the population to support a hypothesis we had previously presented. There were two previous observations we were attempting to explain: (1) the GA performs at least as well in some dynamic environments as it does in static environments, and (2) that the GA performs better in sharply transitioning environments than it does in smoothly transitioning environments. We had one hypothesis that we used to explain these observations. Our hypothesis was that in both static environments and smoothly transitioning environments the GA gets stuck on local optima. Our results show that the evolution of schemata in both of these environments is very similar which lends credence to this hypothesis.

Since the sl-hdf is made up of schemata, by seeing what schemata the GA acquires on average we find out if it is stuck on local optima. If the GA does not acquire any new schemata that means that it is stuck in a place where it can not acquire any new schemata. Thus a local optima can be conceptualized as a place where it is difficult for the GA to make progress, since in the sl-hdf the way the GA makes progress is by acquiring new schemata, by definition then we have shown that the GA is stuck on local optima in these situations. This is some of the best evidence that we can have that the "local optima" hypothesis is true. We could construct a fitness landscape and predetermine where all of the local minima and maxima were, but we would only be able to do that for a particular problem, and thus those results would not be generalizable. By using semantic analysis of the GA's population on the sl-hdfs, we are able to make statements about a larger class of problems.

Though this new analysis has helped us to solve some of the "mysteries" that we presented earlier, it has inevitably uncovered some new mysteries. For instance, why is the acquisition of potholes non-monotonic at times in the Cliffs variant and not in the other two variants, and why is this only seen at the end of the run? The other two variants should also be avoiding areas of negative fitness contribution, and the number of generations should be irrelevant. The shape of the "spikes" in the diversity graph is also very interesting. In both the Cliffs and the Smooth variants, they have a big rise, then a fall, then a slow rise. The size of the spikes also increases until the optimal solution is found. Moreover, overall schemata diversity peaks around the time the population has found an optimal solution. Future research is warranted to try to answer these questions.

Studying the GA on constructed problems allows us to see how it behaves both in terms of performance but also in terms of content and semantics. Moreover, semantic analysis on known problems allows us to offer additional proof to support hypotheses about the operation of the GA. These hypotheses can help practitioners in applying the GA to real world problems and theorists attempting to understand the behavior of the GA.

Acknowledgments. We thank the University of Michigan's Center for the Study of Complex Systems (CSCS) for providing computational resources and support for RR. We also thank the Northwestern Institute on Complex Systems (NICO) for providing support for WR.

References

1. Holland, J.: Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann Arbor, MI (1975)
2. Holland, J.H.: Building blocks, cohort genetic algorithms, and hyperplane-defined functions. Evolutionary Computation **8** (2000) 373–391
3. Rand, W., Riolo, R.: Shaky ladders, hyperplane-defined functions and genetic algorithms: Systematic controlled observation in dynamic environments. In Rothlauf, F., et al, eds.: Evoworkshops. Volume 3449 of Lecture Notes In Computer Science., Springer (2005)
4. Rand, W.: Controlled Observations of the Genetic Algorithm in a Changing Environment: Case Studies Using the Shaky Ladder Hyperplane-Defined Functions. PhD thesis, University of Michigan (2005)
5. Rand, W., Riolo, R.L.: The effect of building block construction on the behavior of the ga in dynamic environments: A case study using the shaky ladder hyperplane-defined functions. In: EvoWorkshops. (2006) 776–787
6. Rosca, J.P.: Analysis of complexity drift in genetic programming. In Koza, J.R., et al, eds.: Genetic Programming 1997, Stanford University, CA, USA, Morgan Kaufmann (1997) 286–294
7. Toffolo, A., Benini, E.: Genetic diversity as an objective in multi-objective evolutionary algorithms. Evolutionary Computation **11** (2003) 151–67
8. Branke, J.: Evolutionary Optimization in Dynamic Environments. Kluwer (2001)
9. Rand, W., Riolo, R.: Measurements for understanding the behavior of the genetic algorithm in dynamic environments. In Beyer, H.G., et al, eds.: GECCO, New York, ACM Press (2005)

Simultaneous Origin-Destination Matrix Estimation in Dynamic Traffic Networks with Evolutionary Computing

Theodore Tsekeris[1,2], Loukas Dimitriou[2], and Antony Stathopoulos[2]

[1] Center of Planning and Economic Research,
Amerikis 11, 10672 Athens, Greece
tsek@kepe.gr
[2] Department of Transportation Planning and Engineering, School of Civil Engineering,
National Technical University of Athens, Iroon Polytechniou 5, 15773 Athens, Greece
lucdimit@central.ntua.gr, astath@transport.ntua.gr

Abstract. This paper presents an evolutionary computing approach for the estimation of dynamic Origin-Destination (O-D) trip matrices from automatic traffic counts in urban networks. A multi-objective, simultaneous optimization problem is formulated to obtain a mutually consistent solution between the resulting O-D matrix and the path/link flow loading pattern. A genetically augmented microscopic simulation procedure is used to determine the path flow pattern between each O-D pair by estimating the set of turning proportions at each intersection. The proposed approach circumvents the restrictions associated with employing a user-optimal Dynamic Traffic Assignment (DTA) procedure and provides a stochastic global search of the optimal O-D trip and turning flow distributions. The application of the model into a real arterial street sub-network demonstrates its ability to provide results of satisfactory accuracy within fast computing speeds and, hence, its potential usefulness to support the deployment of dynamic urban traffic management systems.

Keywords: Evolutionary Computing, Transportation Networks, Origin-Destination Matrices, Traffic Flows, Microscopic Simulation.

1 Introduction

The time-varying (dynamic) Origin-Destination (O-D) trip matrices provide a crucial input for the simulation, management and control of urban road transportation networks. A dynamic O-D matrix specifies the aggregate demand for trip interchange between specific traffic zones of the network over a series of time intervals. The dynamic O-D matrix estimation is typically based on two sources of information, i.e. measured traffic flow time series (counts) at selected network links and a prior O-D matrix to guide the solution procedure. The resulting O-D matrices are mostly used as input to a Dynamic Traffic Assignment (DTA) procedure for mapping the estimated trip demand into a set of path and link traffic flows. In most of the traditional approaches that have been proposed in the literature (see [1]), such a mapping is considered as fixed during the estimation process, based on the assignment of the

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 668–677, 2007.
© Springer-Verlag Berlin Heidelberg 2007

prior O-D matrix onto the network. Namely, the relationship between travel demand and link volumes, as defined by the fractions of each O-D trip population traversing the observed links, known as link use proportions, which constitute the assignment matrix, is held constant. In this case, the problem of inconsistency appears between the initial assignment matrix, on which the estimation process relies, and the final assignment matrix resulted from the estimated demand pattern.

The advent of bi-level programming approaches aimed to address this inconsistency, through the combined estimation of two separate optimization problems, i.e., those of O-D matrix estimation and traffic assignment [2]. In contrast to the iterative (sequential) optimization approach [3], which treats O-D matrix estimation and DTA as two separate problems, that of *simultaneous* (single-level) optimization formulates both the O-D matrix and DTA problems within a single (composite) objective function and it can ensure a mutually consistent solution between the resulting demand and path and link loading pattern in each updating iteration [4]. The latter approach is concisely referred to here as simultaneous dynamic O-D matrix estimation. Nevertheless, the solution of the DTA problem involves an increased computational complexity in realistic urban networks, which restricts the applicability of the simultaneous O-D matrix estimation for practical traffic operation purposes. Such purposes can primarily involve the online deployment of real-time route guidance information and area-wide traffic signal control systems. Also, the problems of inaccuracy typically encountered in the DTA procedure, uncertainty of prior demand information and missing flow information hinder the effort of replicating actual traffic conditions. The usage of suitable, stochastic global search techniques, such as Genetic Algorithms (GAs), to address the above issues involved in O-D matrix estimation is particularly limited in the literature and restricted to the case of static traffic networks [5], [6].

In order to address the aforementioned problems, the present study describes an evolutionary computing approach for the non-DTA-based solution of the simultaneous O-D matrix estimation with automatic link flow measurements in dynamic urban traffic networks. The DTA procedure is substituted here with a series of turning proportion estimation subproblems at the level of each intersection (node). Each of these subproblems is solved with a GA-based algorithm developed in [7]. This paper extends this procedure to carry out a stochastic global search and simultaneous estimation of the optimal distribution of turning movement flows at each intersection and O-D trip flows at the whole network. The modeling of the network traffic flow characteristics is achieved by using a microscopic simulation procedure at the level of individual vehicle, which is integrated with the GA. The computational efficiency of the approach is practically demonstrated through implementation into a real signalized arterial sub-network. Section 2 describes the problem of the simultaneous dynamic O-D matrix estimation and presents the microscopic traffic simulation procedure and the genetic algorithm. Section 3 provides information about the experimental setup of the study. Section 4 presents the computational results obtained from the model implementation and Section 5 summarizes and concludes.

2 Evolutionary Computing Approach

2.1 The Simultaneous Dynamic O-D Matrix Estimation Problem

The simultaneous optimization approach has been considered to address a number of transportation network problems, which can be provided through a bi-level programming formulation (see [8]), including O-D trip matrix estimation. The present formulation ensures a mutually consistent solution between the resulting O-D matrices, link flows and intersection turning proportions at a one-step process. Consider a network composed of G nodes, E directed links, L and M be the amount of origins and destinations from and to which vehicular trips are allocated and W be the amount of O-D pairs traversed by flows. Let $\mathbf{x}$ be the (unknown) O-D trip matrix whose elements $\tilde{x}_{lm}^{\tau}$ denote the number of vehicular trips departing from origin l to destination m during estimation interval $\tau \in T$ and contributing to the flows traversing links during count interval $t \in \tau$, where T is the study period that typically refers to the (morning or afternoon) peak travel period. The trip demand $\tilde{x}_{lm}^{\tau}$ gives rise to path flows p_{lm}^{τ} between each $l-m$ pair. Also, consider J be the total number of observed links, which are equipped with a traffic counter and are traversed by flows that have exited from the upstream node $g \in G_{lm}^{*}$, with G_{lm}^{*} be the set of nodes traversed by the feasible O-D paths between $l-m$ pair, and h_{lmi}^{t} be the flows between $l-m$ pair traversing a link ending at node entrance i during t. Moreover, y_j^t and $\tilde{y}_j^t$ denote respectively the variables of the measured and estimated (assigned) flows traversing the observed link j during count interval t and b_{ij}^{lmt} are the proportions of flow between $l-m$ pair turning from entrance i to link j at interval t.

Then, the simultaneous dynamic O-D matrix estimation at an interval τ can be expressed as a multi-objective optimization problem, through minimizing a composite function F, i.e., a weighted function composed of the Mean Absolute Relative Error (*MARE*) of the resulting O-D trip matrix and link flow estimates, as follows:

$$\min F(\mathbf{x}, \tilde{\mathbf{y}}, \mathbf{b}) = \left\{ \gamma_1 \left[\frac{1}{W} \sum_l^L \sum_m^M \frac{\left| x_{lm}^{\tau} - \tilde{x}_{lm}^{\tau} \right|}{x_{lm}^{\tau}} \right] + \gamma_2 \left[\frac{1}{Z} \sum_{t \in \tau} \sum_j^J \frac{\left| y_j^t - \tilde{y}_j^t \right|}{y_j^t} \right] \right\}, \qquad (1)$$

subject to the constraints:

$$\tilde{x}_{lm}^{\tau} = \sum_l^L \sum_m^M p_{lm}^{\tau} \ , \qquad (2)$$

$$p_{lm}^{\tau} = \sum_t^{\tau} \sum_g^{G_{lm}^{*}} \sum_i^I h_{lmi}^{t} \ , \qquad (3)$$

$$h_{lmi}^{t} = \sum_{l}^{L} \sum_{m}^{M} \sum_{k}^{K} \tilde{y}_{k}^{t} \quad , \tag{4}$$

$$\tilde{y}_{j}^{t} = \sum_{l}^{L} \sum_{m}^{M} \sum_{i}^{I} b_{ij}^{lmt} \, h_{lmi}^{t} \quad , \tag{5}$$

$$\sum_{j}^{J} \tilde{y}_{j}^{t} \le \sum_{k}^{K} \tilde{y}_{k}^{t} \quad , \tag{6}$$

$$0 \le b_{ij}^{lmt} \le 1 \; , \tag{7}$$

$$\sum_{k \ne i}^{K} b_{ik}^{lmt} = 1 \; , \tag{8}$$

The scalar Z in function F is given as $Z = J_T \times V$, with J_T be the sum of all links traversed by observed flows that have exited from an upstream node and V be the traffic flow variables (e.g., volume, occupancy, speed) measured by traffic counters. The index k denotes any (observed or unobserved) link traversed by flows that have exited from a node $g \in G_{lm}^{*}$, with $K \ge J$ be the total number of links traversed by flows exited from that node. The reliability weights γ_1 and γ_2 express the relative confidence assigned to each of the two sources of information, i.e. the prior O-D matrix, with elements x_{lm}^{τ} (see Section 3), and the traffic counts respectively, on the performance of the estimation process. The objective function (1) incorporates the effects of changes in the amount of O-D pairs and traffic counters, the size of O-D demand and measured traffic flow, and the type of traffic flow variable. The equation constraint (2) imposed on O-D trip flows $\tilde{x}_{lm}^{\tau}$, equation constraints (3)-(6) imposed on link traffic flows $\tilde{y}_{j}^{t}$ and physical constraints (7)-(8) imposed on turning flow proportions b_{ij}^{lmt} ensure the simultaneous production of a mutually consistent solution between these three variables in each set of count intervals t wherein an estimation interval τ is partitioned.

The above problem formulation circumvents the need for estimating the dynamic user equilibrium (UE) conditions pertaining to the DTA procedure, which are related to the equalization of the travel costs (times) experienced by travelers in all used paths of each $l-m$ pair. In particular, the estimation of the b_{ij}^{lmt} proportions in each node $g \in G_{lm}^{*}$ allows the endogenous construction of the travel path flows and, hence, the trip demand between each $l-m$ pair, without relying on the degree to which the UE-based paths connecting the various O-D pairs contribute to the observed flow at link j, as occurs in the DTA-based methods. The turning proportion estimation and the endogenous path construction are achieved here through a dynamic non-equilibrium procedure of microscopic traffic simulation augmented with a genetic algorithm, as they are described respectively in the following two subsections.

2.2 The Microscopic Traffic Simulation Procedure

The present study employs a microscopic simulation model to enable the detailed representation of the urban traffic flow characteristics at the level of each vehicle. The simulation logic of the model lies on the similarity between traffic flow and moving particles and the use of simplified rules to represent the movement and interaction between vehicles. Such an approach can more efficiently handle the information requirements of dynamic traffic operations in urban networks of realistic size, in comparison to the coarse use of macroscopic and analytical traffic assignment models (see [9]). The current approach involves a well-documented traffic micro-simulation procedure for tracking the spatial and time trajectories of individual vehicles, based on a car-following (acceleration/deceleration) model of collision avoidance logic, as coded in the NETSIM platform [10]. The model also performs detailed representation of other behavioral features of drivers, such as lane changing and gap acceptance.

The use of such a model can better address the complexity pertaining to the operational and geometric characteristics and the underlying traffic dynamics of signalized arterial networks. In particular, it enables the coding of such network characteristics as signalized traffic control strategies, transit operations, pedestrian and parking activities. The loading of the network is carried out through providing information about the flows at the entry nodes, including the origin and destination zones, and the turning proportions at each node. The resulting path flow pattern can involve the use of alternative feasible routes between an O-D pair, which intrinsically reflects the existence of different cost perceptions of travelers. This stochastic dispersion of path flows can be considered as a more realistic route choice assumption, in comparison to that relying on the deterministic (shortest path-based) behavior.

2.3 Description of the Genetic Algorithm

The present optimization problem, as described in 2.1, is convex and, hence, it may ensure a feasible and unique solution for the O-D trip matrix estimation process. Nonetheless, the intricate nature of the solution procedure, principally due to the increased dimensionality, complexity of the search space and existence of many local minima in realistic urban networks, imposes an uncertainty in reaching an optimal (or satisfactory sub-optimal) solution. The use of evolutionary computing techniques, such as the GA presented here, can address this uncertainty by relaxing the solution dependencies on the availability, quality and variability of the prior aggregate demand and traffic count information, in contrast to the traditional greedy search algorithms. Moreover, the event-based simulation procedure (see 2.2) employed for processing the movement trajectory of each individual vehicle makes the problem highly complex and non-linear so that necessitates the application of evolutionary computing techniques, instead of using gradient-based methods which commonly appear in the existing literature (see Section 1).

The operations that correspond to the GA resemble the natural processes of the evolution of a population, i.e. reproduction, crossover and mutation, as they are analytically discussed in the relevant literature (see [11]). The following paragraphs focus on describing the GA processes carried out in the current application. The GA

utilizes a set (population) of strings called chromosomes, each representing a feasible matrix of b_{ij}^{lmt} proportions and, consequently, a feasible solution to the problem. Every entry in the chromosome is called allelic value and the representation of the chromosomes and the assignment of the allelic values are based on a binary {0, 1} coding scheme (see Section 3).

The setup of the GA initialization relies on ensuring a favorable tradeoff between convergence speed and population size, as it is described in [11]. The members of the initial population are created by random perturbation, across a specific range, of those b_{ij}^{lmt} values obtained through the simulation-based assignment of the prior O-D matrix onto the network. The solution procedure involves the repeated execution of two stages: (i) the calculation of a fitness function, i.e. objective function (1), for each individual, based on the results of the micro-simulation model, and (ii) the production of a 'genetically improved' population of turning proportions at each count interval $t \in \tau$. This procedure takes place at each node $g \in G_{lm}^{*}$ until achieving a satisfactory level of accuracy at the whole network (see Section 3).

The reproduction operator uses a tournament selection between three candidate individuals. The crossover operation enables the exchange of genetic information between the old population members in order to obtain the new ones. This exchange is carried out by randomly selecting slots among the strings and swapping the characters between the selected positions. A relatively high crossover rate is selected here (70%), which enhances the probability of the selected individuals to exchange genetic information. A mutation operator is used to prevent an irrecoverable loss of potentially useful information, which can be occasionally caused by reproduction and crossover and, hence, it reduces the probability of finding a false peak. This operator provides an occasional random alteration of the allelic value of a gene with a small probability, which is here set equal to 5%.

3 Experimental Setup

The area of the present application corresponds to a part of the urban road network of Athens, Greece. This area is located in the periphery of the city centre and covers a major arterial sub-network (Alexandras Avenue), which is controlled with a fixed time signal strategy (1.5 Generation). The specific arterial network (see Fig. 1) is composed of 46 links and 32 nodes, 16 of which are entry-exit nodes and 16 are internal nodes. Hence, the dimensions of the O-D matrix are 16x16 and those of the matrix of turning proportions are 46x4, where 4 corresponds to the total amount of possible turning movements, i.e. left, through, right and diagonal. Due to the absence of diagonal movements and prohibitions on several left turning movements, the number of (unknown) turning proportions reduces from 46x4=184 to 66. The GA population is composed of 50 individuals and the coded values for each turning movement lie between 255 (11111111) and 1 (00000001), since a number of 8 alleles is adopted to correspond to each link and turning proportion at each node. Thus, the length of each chromosome is 66x8=528 alleles. Finally the values of the turning movements are transformed to turning flow proportions.

The estimation process utilizes loop detector measurements corresponding to entry flows at the 16 entry-exit nodes and to link traffic flows traversing 4 selected sections along the major arterial, for both flow directions, i.e. J_T =8. The link flow measurements refer to both smoothed volume and occupancy values, namely, V =2. These four measurement sections, whose location is shown by the markings of Fig. 1, facilitate the constant monitoring of traffic operating conditions along the entire length of the sub-network. The traffic data are collected at the end of every 90-sec signalization cycle and are transmitted to the Traffic Control Center of the city. The traffic measurements of the current dataset are aggregated at 3-min count intervals and correspond to a typical peak hour of the morning travel period of the day. A time-partitioned (partial) O-D matrix corresponds here to a typical estimation interval τ of 15 min, since such a duration can be considered as adequate for a user to traverse the given part of the network.

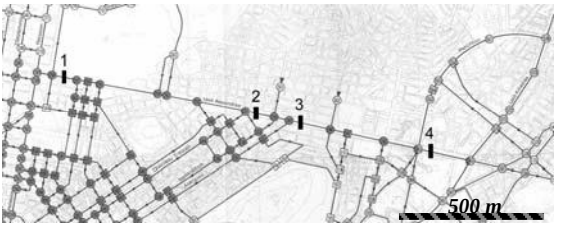


Fig. 1. The layout and coding of Alexandras Avenue sub-network

The convergence (or termination) of the GA is based on two different empirical criteria. The first criterion refers to the estimation accuracy of the solution procedure, which is set equal to a small value of objective function F =0,05 (or 5%). The second (stopping) criterion is related to the intended practical usage of the model. The current application concerns the real-time deployment of an area-wide network traffic monitoring and control system. For this reason, each partial O-D matrix is regularly updated on a rolling horizon framework, according to the frequency of collecting and processing traffic flow information. In the present study, this frequency refers to a count interval t of 3 min. Thus, a maximum running time of 3 min is set as the stopping criterion for the purposes of the specific application.

The prior O-D matrix was synthesized through the offline implementation of the proposed simultaneous optimization process by using a set of ‘typical’ (average) traffic flows over the observed links for the given study period. These ‘typical’ flows were obtained from averaging a series of volumes and occupancies corresponding to the specific period-of-the-day of the past four weeks. This study uses the assumption of the similar reliability and equal contribution of the prior demand and traffic count information on the performance of the estimation process, i.e. $\gamma_1 = \gamma_2$ =0,5. The O-D trip flows are expressed here in the form of trip rates, as obtained by the ratio of the trip demand across a specific O-D pair to the total demand size in the network. Such a transformation helps relaxing the dependency of the solution on the scale of demand.

4 Computational Results

The performance of the estimation procedure is investigated with regard to both the solution accuracy and the convergence speed, based on the convergence criteria described in Section 3. The micro-simulation of the present network processed a total amount of approximately 2000 vehicle-trips during the given study period. The GA, which was coded in FORTRAN 90/95 workstation, is found to reach a stable solution with the required level of accuracy, i.e. F =5%, in an average computing duration of less than 3 min, which corresponds to about 2000 runs (or 40 generations).

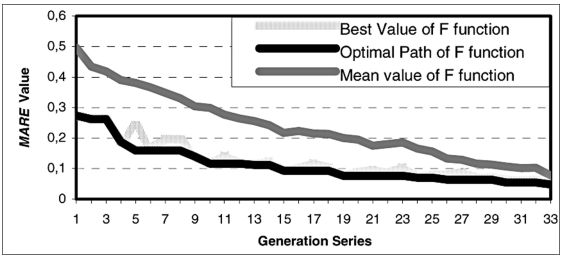


Fig. 2. Results of the objective function F of the GA

In particular, the typical convergence behavior of the GA involves a number of 20-30 generations, while a maximum number of 40 generations is only required for cases of population members of poor quality, i.e. with very small initial fit (less than 5% of the cases). Fig. 2 indicates the best value and the mean value of the F function in each generation as well as its optimal path across the series of generations for the first estimation interval of the study period. The graph shows that the GA performance, in terms of the improvement (reduction) of the F function, is steadily increasing in the first 20 generations, while convergence is achieved after 30 generations. Fig. 3 illustrates the convergence behavior of the two different components of the objective function, i.e. those of the O-D trip matrix and link flow estimation. The graph indicates a tradeoff mechanism between the *MARE* of the two components across generations. Nonetheless, the *MARE* of both components reduces considerably, in the context of the given application, reaching finally a similar level of accuracy, as GA approaches to convergence. Specifically, the *MARE* of the O-D matrix estimation component presents an average reduction of 77%, while the *MARE* of the link flow estimation component presents an average reduction of 88%. This outcome signifies the fact that the mutually consistent solution achieved through the simultaneous optimization process does not lead to the loss of accuracy of the resulting O-D matrix, in favor of the link flow solution accuracy, as it typically occurs in the case of the iterative optimization process [3]. In particular, the average improvement of the accuracy of the link flow solution, in terms of the *MARE*, is found to exceed 75% for all measurement sections considered in the study. Fig. 4 presents, for demonstration purposes, the GA convergence procedure of the link flow estimation subproblem corresponding to four different measurement sections along the major arterial.

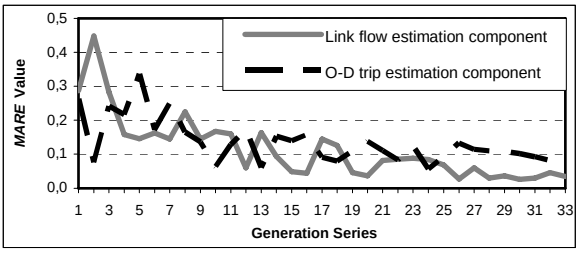


Fig. 3. Results of the O-D matrix and link flow estimation components of the objective function

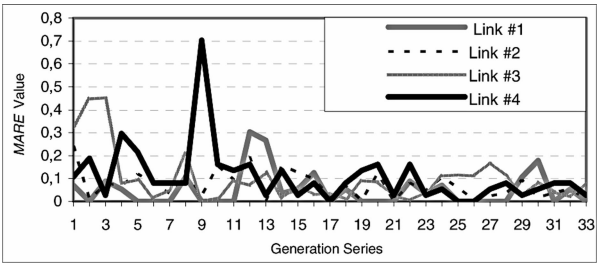


Fig. 4. Convergence results of the link traffic flow estimation at four measurement sections

5 Conclusions

This paper presented a method for solving the problem of the simultaneous dynamic O-D matrix estimation by use of evolutionary computing. The current approach can ensure the production of a mutually consistent solution between the O-D matrix and the path/link flow loading pattern, circumventing the use of a DTA procedure. In particular, this approach relies on the real-time estimation of turning proportions at each intersection, employing information about the prior O-D matrix structure, measurements on entry node flows and link traffic flows, and network loading information obtained from a micro-simulation model.

The GA is found to provide substantial gains to both the O-D matrix and link flow estimation accuracy, in comparison to the initial solution based on the prior demand information. In addition, the computing speed of the algorithm can be considered as satisfactory, taking into account the complexity and processing requirements of the micro-simulation model and the realistic size of the subarea network. The high accuracy of both the resulting O-D trip matrix and link traffic flows, in conjunction with the fast computing speed, suggest the potential of implementing the proposed procedure for purposes of real-time monitoring of realistic urban network conditions. In addition, the present approach can be used as a prototype to support a range of other dynamic traffic management operations. Such operations can include the deployment and evaluation of signal plan coordination and route guidance strategies at specific parts of urban-scale road networks.

References

1. Cascetta, E., Nguyen, S.: A Unified Framework for Estimating or Updating Origin/Destination Matrices from Traffic Counts. Transp. Res. 22B (1988) 437-455
2. Yang, H., Sasaki, T., Iida, Y., Asakura, Y.: Estimation of Origin-Destination Matrices from Link Traffic Counts on Congested Networks. Transp. Res. 26B (1992) 417-434
3. Tavana, H., Mahmassani, H.: Estimation of Dynamic Origin-Destination Flows from Sensor Data using Bi-Level Optimization Method. In: Transportation Research Board Annual Meeting, National Research Council, Washington, D.C. (2001)
4. Van der Zijpp, N.J., Lindveld, C.D.R.: Estimation of Origin-Destination Demand for Dynamic Assignment with Simultaneous Route and Departure Time Choice. Transp. Res. Record 1771 (2001) 75-82
5. Kim, H., Baek, S., Lim, Y.: Origin-Destination Matrices Estimated with a Genetic Algorithm from Link Traffic Counts. Transp. Res. Record 1771 (2001) 156-163
6. Stathopoulos, A., Tsekeris, T.: Hybrid Meta-Heuristic Algorithm for the Simultaneous Optimization of the O-D Trip Matrix Estimation. Comp.-Aided Civ. Infrastr. Engrg. 19 (2004) 421-435
7. Dimitriou, L., Tsekeris, T., Stathopoulos, A.: Genetic-Algorithm-Based Micro-Simulation Approach for Estimating Turning Proportions at Signalized Intersections. In: van Zuylen, H., Middelham, F. (eds.): Proc. 11th IFAC Symposium on Control in Transp. Systems, Delft, The Netherlands (2006) 159-164
8. Yang, H., Bell, M.G.H.: Transport Bilevel Programming Problems: Recent Methodological Advances. Transp. Res. 35B (2001) 1-4
9. Ziliaskopoulos, A.K., Peeta, S.: Review of Dynamic Traffic Assignment Models. Networks Spat. Econ. 1 (2001) 233-267
10. Federal Highway Administration: Traffic Network Analysis with NETSIM- A User Guide. Federal Highway Administration, U.S. Dept of Transportation, Washington, D.C. (1980)
11. Goldberg, D.E.: Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley, Reading, MA (1989)

Evolutionary Combinatorial Programming for Discrete Road Network Design with Reliability Requirements

Loukas Dimitriou[1], Theodore Tsekeris[1,2], and Antony Stathopoulos[1]

[1] Department of Transportation Planning and Engineering, School of Civil Engineering, National Technical University of Athens, Iroon Polytechniou 5, 15773 Athens, Greece
lucdimit@central.ntua.gr, astath@transport.ntua.gr
[2] Center of Planning and Economic Research, Amerikis 11, 10672 Athens, Greece
tsek@kepe.gr

Abstract. This paper examines the formulation and solution of the discrete version of the stochastic Network Design Problem (NDP) with incorporated network travel time reliability requirements. The NDP is considered as a two-stage Stackelberg game with complete information and is formulated as a combinatorial stochastic bi-level programming problem. The current approach introduces the element of risk in the metrics of the design process through representing the stochastic nature of various system components related to users' attributes and network characteristics. The estimation procedure combines the use of mathematical simulation for the risk assessment with evolutionary optimization techniques (Genetic Algorithms), as they can suitably address complex non-convex problems, such as the present one. The implementation over a test network signifies the potential benefits of the proposed methodology, in terms of intrinsically incorporating stochasticity and reliability requirements to enhance the design process of urban road networks.

Keywords: Network Reliability, Stochastic Discrete Network Design, Game Theory, Mathematical Simulation, Genetic Algorithms.

1 Introduction

The role of the design of transportation networks has been recognized as a crucial element to foster the mobility of people and goods in growing metropolitan areas. The conflicting goals of saving scarce resources, such as land and public funds, allocated to road infrastructure and accommodating the increased demand for passenger and freight transportation prompt the need for deploying compromise and efficient design solutions. In addition to the supply of adequate capacity, the volatile traffic conditions caused by recurrent congestion phenomena and incidents amplify the complexity of the design problem, since they raise risk concerns with regard to the reliability of the provided transportation services. Further risk concerns also emerge from the need for making urban transportation networks capable to provide sufficient lifelines during unexpected, emergency situations, due to man-made or physical disasters.

The question of determining the optimum network design, typically referred to as the Network Design Problem (NDP), can be traditionally addressed through two

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 678–687, 2007.
© Springer-Verlag Berlin Heidelberg 2007

different mathematical forms (for general reviews, see [1], [2]). The first form refers to the Continuous-NDP (C-NDP), where the capacity of the system is treated as a continuous variable and can be expressed in terms of vehicles, passengers and unit loads. The second form refers to the Discrete-NDP (D-NDP), which is formulated in terms of discrete (integer or binary) variables, such as the number of new links, in the case of network expansion, or the number of lane additions, in the case of network enhancement etc. Despite that current research on the NDP is particularly active, the majority of existing studies deals with the continuous form, which can be regarded as a relaxation of the discrete one. Moreover, although several reliability considerations have been incorporated into the structure of the C-NDP (see [3], [4], [5]), no such attempt has been made for the case of the D-NDP.

The present study addresses the reliable D-NDP, i.e. the D-NDP with reliability requirements. The solution of such a problem is infrastructure related, since it corresponds to the number of added lanes and new links, which may have a greater bearing to designing the required civil works, in comparison to the solution of the corresponding C-NDP. The total travel time reliability is considered here as a network quality performance indicator, since it reflects the ability of the network to respond to different states of the system. The study provides a formulation and a solution algorithm for the reliable D-NDP, whose application is illustrated for a simplified network with typical urban road settings. Section 2 presents the formulation of the reliable D-NDP. Section 3 analyzes the components of the network reliability and describes a simulation method for performing the risk assessment. Section 4 presents an evolutionary algorithm for the efficient solution of the complex D-NDP. Section 5 includes the results obtained from the application of the method into the test network and Section 6 concludes.

2 Formulation of the Reliable D-NDP

Similar to many other transportation planning problems, the network design process is essentially affected by decisions made on multiple hierarchical levels, concerning both the demand and supply properties of the system [6]. The design process of transportation networks (system) is regarded as a game among two players namely the system designer and the system users, whose decisions made individually affects both their performance. The particular structure of the above game has the form of a two-stage leader-follower Stackelberg game with perfect information, with the system designer to be the leader imposing modifications on the network attempting the optimization of the system performance while the users reacting as followers to alternative design plans. The formulation of such games is usually has the form of bi-level programming problems, where optimum strategies are sought by taking into account a number of constraints, including those of physical feasibility and budget availability, while considering the demand and supply attributes of the system as known but not necessarily fixed. This study extends the standard game-theoretic, bi-level programming formulation of the D-NDP so that include reliability requirements.

Consider a network composed of L links, R origins and S destinations. The travel demand q_{rs} gives rise to equilibrium flows f_k^{rs} along path $k \in K_{rs}$ connecting

$r-s$ pair and to equilibrium flows $x_a(y)$ along link a, with δ_{ak}^{rs} be the path-link incidence variable and $c_{rs}^{k} \in C^{rs}$ be the cost of traveling along the k th path between $r-s$ pair. The travel cost, at equilibrium state, of some link a with capacity y_a is denoted as $c_a(x_a(y))$, with y be the total network link capacity, and w_a is a binary decision variable of link a, as follows: *1* if link a is added to the network or an extra lane is added to link a, and *0* otherwise. Also, $V_a(w_a)$ denotes the monetary expenditures for adding a link a or a lane on link a, B is the total available construction budget for network capacity improvement, and θ is a factor converting monetary values to travel times. Then, the *Upper-Level Problem*, which comprises the objectives of the optimum NDP, and the *Lower-Level Problem*, which provides the path and link equilibrium flows, can be given as follows:

Upper-Level Problem:

$$\min_{y} F(x,y) = \sum_{a \in A} \left(E[c_a(x_a(y), w_a) x_a(y)] + \theta V_a(w_a) \right) \tag{1}$$

$$\text{Subject to} \qquad w_a \in \{0,1\}, \qquad \forall\, a \in A \tag{2}$$

$$\sum_{a \in A} (V_a(w_a)) \le B, \qquad \forall\, a \in A \tag{3}$$

$$P\left(\sum_{a \in A} (c_a(x_a(y), w_a)\, x_a(y)) \le T\right) \le Z, \quad \forall\, a \in A \tag{4}$$

Lower-Level Problem:

$$\min_{x} G(x) = -\sum_{rs} q_{rs}\, E\left[\min_{k \in K_{rs}} \left\{ c_{rs}^{k} \right\} \middle| C^{rs}(x) \right] + \sum_{a} x_a c_a(x_a) - \sum_{a} \int_{0}^{x_a} c_a(\omega) d\omega \tag{5}$$

$$\text{Subject to} \qquad f_k^{rs} = P_k^{rs}\, q_{rs}, \qquad \forall \quad k, r, s \tag{6}$$

$$x_a = \sum_{rs} \sum_{k} f_k^{rs}\, \delta_{ak}^{rs}, \quad \forall\, a \in A \tag{7}$$

$$f_k^{rs},\, x_a \ge 0, \qquad \forall\, a \in A \tag{8}$$

In the Upper-Level Problem, $F(x,y)$ represents the objective function of the NDP, wherein the first component refers to the travel cost expressed in terms of the expectation E of network *Total Travel Time* (*TTT*), and the second component corresponds to the total expenditures (in time units) for capacity improvements. The choice set defined in relationship (2) represents the binary selection of link (or lane) addition, while inequality (3) imposes budgetary restrictions. The reliability requirements are introduced in constraint (4), through restricting the probability of the *TTT* to be lower than or equal to a pre-specified upper limit T, with Z defining the

acceptable confidence interval ($0 \le Z \le 1$) for this hypothesis. Such a condition essentially depicts the stability of the system [7].

The Lower-Level Problem, which consists of functions (5) to (8), performs the trip demand assignment process, based on the expected (perceived) value E of the path travel cost c_{rs}^{k}. Specifically, it estimates the response of users to the capacity improvements made at the Upper-Level Problem, through determining the probability P_k^{rs} that a traveler chooses to use path k between $r-s$ pair. The Stochastic User Equilibrium (SUE) model [8] is used here for the assignment of demand onto the network and the solution Method of Successive Averages (MSA) is employed to calculate equilibrium flows.

3 Modeling Reliability Assessment in the D-NDP

The operational performance of transportation systems typically relies on variables of uncertain nature, as their values are influenced by random events and human decision-making processes. The risk involved in the operation of transportation networks can be mainly attributed to the uncertainty pertaining to four different components: the demand, the supply (capacity), the level of service (link travel time) and the users' characteristics (route choice behavior). By and large, travel demand patterns in urban transportation networks can be considered as recurrent, at typical operating conditions. Nonetheless, several disturbances can be observed, as expressed by seasonal or random spatial and temporal variations of demand between origins and destinations, due to special (planned or unexpected) events in different network localities, causing significant fluctuations in link travel times. These disturbances are additionally influenced by several characteristics of travelers, which are mostly related to factors involved in the route choice decision-making process, including perception of travel cost, value of travel time and driving behavior. Capacity fluctuations is also a typical phenomenon in transportation networks, which can emerge from several factors, such as changes in the composition of traffic, congestion effects and other, random phenomena, like accidents, road workzones and adverse weather conditions. The problem of network reliability degradation considering fluctuations in link capacities has been extensively investigated in the literature (see [9], [10], [11]).

The current study provides a mathematical simulation framework for representing the stochastic properties and, in turn, the fluctuations of the values of variables, i.e., demand, supply and travel time, which affect the system performance. In particular, the demand is considered here as a random variable following the normal distribution $N\left(\mu_{rs}, \sigma_{rs}^2\right)$, with μ_{rs} denoting its mean for each $r-s$ pair and σ_{rs}^2 denoting its variance. Despite the fact that travel time can be also described by the normal distribution [7], an alternative, more explanatory assumption is made here, assuming that link speeds follow a multinomial normal distribution correlated with the speeds of the neighboring links. A similar assumption is adopted for the distribution of link capacities.

The current framework connects link travel time to link speed and capacity fluctuations and it enables to express the interaction between the costs of each link. In

this way, link travel time variability (which is the result) is intrinsically modeled, in the structure of the D-NDP, with regard to its causal phenomenon (which is the link capacity and speed variability). More specifically, the Lower-Level Problem enables the estimation of the statistical properties of the *TTT*, i.e. its mean value and variance, which are subsequently fed to the Upper-Level problem, through iterating the solution of the assignment procedure, comprising the set of link and path equilibrium flows, the values of origin-destination demand, link capacity, and the link free flow travel time, in accordance with the stochastic characteristics assigned to these variables, as described previously.

The estimation of the statistical properties of *TTT* and, hence, the reliability assessment, are performed through the simulation method of the *Latin Hypercube* sampling. In comparison to other simulation methods, such as that of Monte Carlo simulation, Latin Hypercube is based on a stratified random procedure which provides an efficient way to capture the properties of the stochastic variables from their distributions, namely, it produces results of higher accuracy without the need for increasing the sampling size, and it allows to model correlations among different variables. In particular, the procedure of Iman and Conover [12] is followed here in order to produce correlated random numbers from the normal distribution, based on the Cholesky decomposition of the correlation matrix. The assumptions concerning the usage of the simulation method in the network design process are:

i) The duration of changes in link speeds and capacities allows users to re-estimate route choices, and
ii) Link speed reduction is due to random events (like accident, physical disaster or other) which affect a locality of the network and, hence, link capacities and speeds are correlated with those of neighboring ones.

4 Evolutionary Algorithm for Solving the Reliable D-NDP

The D-NDP, as well as the C-NDP, can be generally characterized as problems of increased computational complexity. This complexity arises from the fact that bi-level programming problems, even for simple linear cases, are Non-deterministic Polynomial-time (*NP*)-hard problems [13]. In particular, the D-NDP is a *NP*-hard, non-convex combinatorial problem [14], since its set of constraints involves non-linear formulations, such as those of the SUE assignment of the Lower-Level Problem. Several algorithms, appropriate for addressing complex combinatorial (integer or mixed-integer programming) problems, have been hitherto proposed and implemented to solve the D-NDP. Such algorithms include the branch-and-bound method [15], Lagrange relaxation and dual ascent procedures [1] and a method based on the concept of support function [16].

The present study uses evolutionary strategies, in particular, a Genetic Algorithm (GA) [17] to address the difficulties of obtaining optimal solutions within the proposed framework of the D-NDP. GAs have been extensively used in solving complex, *NP*-hard, combinatorial problems. Furthermore, they have been widely used in various bi-level programming problems (see [18]) and, especially, in addressing the C-NDP [19]. This wide applicability of GAs can be attributed to their convenience in handling variables of stochastic nature and multiple constraints in a seamless way,

without requiring information about the nature of the problem but only about the performance of a 'fitness' function for various candidate states.

GAs are population-based stochastic, global search methods. In the context of the D-NDP, an individual of the population corresponds to alternative binary codings of the link and lane additions. For every individual of the population, a Latin Hypercube simulation is performed altering the travel demand, link travel time and capacities, based on the framework described in Section 3, in order to estimate *TTT* reliability. The steps of the solution procedure are given to the pseudo-code shown below:

Step 1. (Initialization)

Produce an initial random population of candidate feasible solutions (link capacity improvements or link additions) and select the properties of the genetic operators

DO UNTIL CONVERGENCE:

Step 2. (Path Enumeration)

Perform path enumeration for every candidate solution

Step 3. (Simulation)

Estimate the *TTT* reliability for every candidate solution by Latin Hypercube simulation.

Step 4. (Genetic Evolution Process)

4.1 Check for the consistency of constraints and estimate the 'fitness function' (1) of each candidate solution.

4.2 Perform a stochastic selection of the 'fittest' solution set and the crossover operation among the selected 'individuals'

4.3 Perform the mutation of individuals

4.4 Produce a new population of genetically improved candidate solutions

Though the extended use of meta-heuristic techniques such as GAs for solving complex problems, the solutions provided by them have met some skepticism. This is mainly because they are heavily dependent on initial conditions and random search processes incorporated into them. In order to confront this problem, multiple runs of the GA are performed in this study to confirm that the solution provided is optimal (or adequately near-optimal).

5 Test Application and Results of the Algorithm

The proposed methodology for solving the reliable D-NDP is implemented into a test network. The specific network layout has been used in [16] and is composed of a single origin-destination pair (from node #1 to node #12), 12 nodes, 17 existing links

and 6 candidate new links. Fig. 1 shows the complete configuration of the test network, including a total of 23 links and 25 paths, after making all possible improvements, i.e. adding the links figured #18-23. The network is considered as fully degradable, in the sense that all links exhibit fluctuations in speed and capacity. The complexity of this specific combinatorial problem, although the small scale of the given network, can be considered as high, since it contains L =23 links (variables), yielding $2^L=2^{23}=8388608$ possible combinations (alternative construction plans) of network capacity improvement.

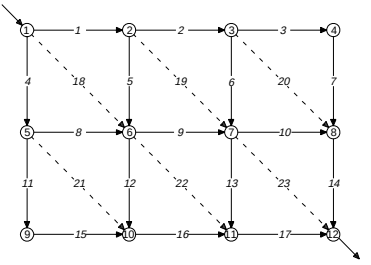


Fig. 1. The test network layout: existing links (*solid lines*) available for lane additions and potential new links (*dashed lines*)

In the current study, the link travel time t_a at some link a is expressed as a function of the random free-flow travel time t_a^f, traffic flow x_a and random capacity y_a at this link and is carried out by using the standard formulation of the Bureau of Public Roads (BPR), as follows:

$$t_\alpha = t_a^f \left(1+\beta \left(\frac{x_\alpha}{y_a} \right)^m \right), \tag{9}$$

where β and m are scale parameters depending on the operational characteristics of the network. Although the estimation of link travel time is based here on the BPR formula, which typically applies to uncongested road networks, other formulations could also be adopted for taking into account congestion effects, like queues or bottlenecks in the links of the network.

The capacity of each of the existing links is set equal to 20 vehicles per hour (veh/hr), while, after a lane addition, the capacity increases to 30 veh/hr. The capacity of each of the new links is set equal to 20 veh/hr. The demand between the origin-destination pair is set equal to 80 veh/hr. The cost of lane addition in each of the existing links is set equal to 30 monetary values, while the construction cost of each of the new links is set equal to 50 monetary values. This study adopts a conversion factor $\theta = 1$. The free-flow travel time, which is proportional to the link length, is set equal to $t_a^f = 1$ min, for the existing links, and $t_a^f = 1.4$ min, for the new links. The complete reconstruction of the given network, which requires a lane addition to each of the 17 existing links and construction of 6 new links, amounts to a total of 810 monetary values, corresponding to 810 vehicle-minutes (veh-min), in time units, since

$\theta = 1$. Nonetheless, such a scenario may be considered as too expensive and, hence, impractical in real-world situations. For this reason, the half of this amount, i.e. 400 veh-min, is set here as the total available construction budget B, which can be regarded as sufficient for enhancing the capacity of the existing network. The estimation of the stochastic user-equilibrium link flows employs a total of 200 MSA iterations, which were found to be adequate for providing a stable solution to the Lower-Level Problem for the given test network.

An initial solution is first obtained by solving the D-NDP without reliability requirements. The assignment of the travel demand onto the initial network (with no link improvements or additions) results in a *TTT* equal to 886 veh-min. A total number of 50 runs of the solution algorithm are performed. The current GA employs a population of 50 individuals and its convergence criterion requires, on average, a number of 50 generations, which ensures that no further significant improvement of the objective function value can be achieved. For each individual, 200 iterations of the Latin Hypercube simulation are performed to obtain the stochastic properties of the system variables (see Section 3). Both the variances of the multinomial normal distributions of the free flow speed and link capacity are set here equal to the 20% of their corresponding theoretical (nominal) mean values. Similarly, the variance of the normal distribution of travel demand is set equal to the 20% of its mean (set) value.

The initial solution provides a construction plan encompassing the addition of the new links #18 and #23 and lane addition to the existing links #1, #9 and #17. The resulting construction cost amounts to 190 veh-min, which is lower than the total available budget (400 veh-min). The *TTT* reduces from 886 veh-min to 449 veh-min. Thus, the total cost (*TTT* + construction cost) comes to 449+190=639 veh-min. The existence of a considerable remaining (not allocated) portion of the total available budget, i.e., 400-190=210 veh-min, can be attributed to the fact that there is a threshold beyond which network capacity improvements become very expensive, in comparison to their contribution to the reduction of the *TTT*.

A new solution of the D-NDP is then obtained by imposing, as reliability requirement, the probability of the occurrence of the *TTT* to be higher than T =500 veh-min not to exceed 10%. This upper limit value expresses the 10% increment of the *TTT* value (449 veh-min) resulted from the initial problem solution. The new solution leads to the construction of two more links, i.e. links #19 and #22, in addition to the improvements resulted from the initial solution of the problem. The *TTT* is further reduced from 449 veh-min to 423 veh-min, while the construction cost is raised to 290 veh-min, which is still less than the total available budget. The raised construction cost is attributed to the need for increasing the amount of the sparse total network link capacity (system redundancy) in order to ensure the desired level of network reliability. The new solution results in the increase of the total cost from 639 veh-min to 423+290=713 veh-min. The resulting probability of the *TTT* to be higher than 500 veh-min was estimated to 9.94%, which is lower than the acceptable upper bound of 10%. As it is shown in the two histograms of Fig. 2, the dispersion of the *TTT* obtained from the initial solution without imposing reliability requirements, having $P(TTT \geq 500\,veh-\min) \approx 0.25$ (see left diagram) is considerably wider than the dispersion of the *TTT* obtained from solving the reliable D-NDP, having $P(TTT \geq 500\,veh-\min) < 0.1$ (see right diagram).

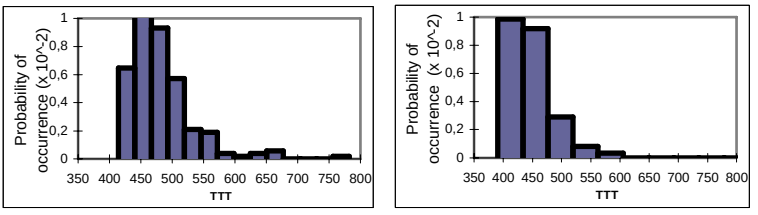


Fig. 2. Distribution of the *TTT* for the case without reliability requirements (*left*) and with reliability requirements (*right*)

6 Conclusions and Future Research

The current study provided a formulation and a solution algorithm for the Discrete-Network Design Problem (D-NDP). The formulation seeks the optimal network capacity improvements, subject to budgetary and physical restrictions, and, additionally, reliability requirements, in terms of the probability of the Total Travel Time (*TTT*) to be less than a pre-specified value. The model enables the inclusion of four different sources of uncertainty, i.e., demand, capacity, link travel time and route choice, into the reliability assessment, through applying the Latin Hypercube sampling simulation method. The estimation procedure uses a Genetic Algorithm, which is suitable for solving such types of stochastic combinatorial optimization problems. The test network application of the method demonstrated the beneficial impact of including reliability requirements into the standard bi-level programming formulation of the D-NDP. The benefits correspond to the reduction of the *TTT*, while satisfying the desired level of network reliability and the budget constraints.

Future developments of the method refer to the incorporation of other types of uncertainty, which affect network reliability, such as those concerning the information acquisition and the day-to-day (or period-to-period) adjustment of the decision-making process of users. Moreover, the current estimation framework could be extended into the more general case of multi-modal networks with multiple-class users, in order to address issues related to the sustainable network development. Finally, a comparison of the current evolutionary approach with other derivative-free algorithms suitable to handle such combinatorial problems could provide useful insight into the properties of the solution of the NDP.

References

1. Magnanti, T.L., Wong, R.T.: Network Design and Transportation Planning: Models and Algorithms. Transp. Sci. 18 (1984) 1-55
2. Friesz, T.L.: Transportation Network Equilibrium, Design and Aggregation: Key Developments. Transp. Reviews 18 (1985) 257-278
3. Waller, S.T., Ziliaskopoulos, A.K.: Stochastic Dynamic Network Design Problem. Transp. Res. Record 1771 (2001) 106-113
4. Yin, Y., Iida, H.: Optimal Improvement Scheme for Network Reliability. Transp. Res. Record 1783 (2002) 1-6

5. Sumalee, A., Walting, D.P., Nakayama, S.: Reliable Network Design Problem: The Case with Uncertain Demand and Total Travel Time Reliability. In: Transportation Research Board Annual Meeting, National Research Council, Washington, D.C. (2006)
6. Fisk, S.C.: A Conceptual Framework for Optimal Transportation Systems Planning with Integrated Supply and Demand Models. Transp. Sci. 20 (1986) 37-47
7. Bell, M.G.H., Iida, Y.: Transportation Network Analysis. Wiley, Chichester (1997)
8. Sheffi, Y.: Urban Transportation Networks: Equilibrium Analysis with Mathematical Programming Methods. Prentice-Hall, Englewood Cliffs NJ (1985)
9. Du, Z.P., Nicholson, A.: Degradable Transportation Systems: Sensitivity and Reliability Analysis. Transp. Res. 31B (1997) 225-237
10. Bell, M.G.H.: A Game Theory Approach to Measuring the Performance Reliability of Transport Networks. Transp. Res. 34B (2000) 533-545
11. Chen, A., Yang, H., Lo, H.K., Tang, W.H.: Capacity Reliability of a Road Network: An Assessment Methodology and Numerical Results. Transp. Res. 36B (2002) 225-252
12. Iman, R.L., Conover, W.J.: A Distribution-Free Approach to Inducing Rank Correlation among Input Variables. Commun. Stat. B11 (1982) 311-334
13. Ben-Ayed, O., Boyce, D.E., Blair, C.E.: A General Bi-Level Programming Formulation of the Network Design Problem. Transp. Res. 22B (1988) 311–318
14. Papadimitriou, C.H., Steiglitz, K.: Combinatorial Optimization: Algorithms and Complexity. Dover Publications, Mineola, NY (1998)
15. Leblanc, L.J.: An Algorithm for the Discrete Network Design Problem. Transp. Sci. 9 (1975) 183–199
16. Gao, Z., Wu, J., Sun, H.: Solution Algorithm for the Bi-Level Discrete Network Design Problem, Transp. Res. 39B (2005) 479-495
17. Goldberg, D.E.: Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley, Reading, MA. (1989)
18. Colson, B., Marcotte, P., Savard, G.: Bilevel Programming: A Survey. 4OR: Quart. J. Oper. Res. 3 (2005) 87-107
19. Cree, N.D., Maher, M.J., Paechter, B.: The Continuous Equilibrium Optimal Network Design Problem: A Genetic Approach. In: Selected Proc. 4th EURO Transportation Meeting, Newcastle (1998) 175-193

Intelligent Traffic Control Decision Support System

Khaled Almejalli, Keshav Dahal, and M. Alamgir Hossain

MOSAIC Research Group, School of Informatics,
University of Bradford, Great Horton Road Bradford, BD7 1DP, United Kingdom
{k.a.al-mejjali,k.p.dahal,m.a.hossain1}@bradford.ac.uk

Abstract. When non-recurrent road traffic congestion happens, the operator of the traffic control centre has to select the most appropriate traffic control measure or combination of measures in a short time to manage the traffic network. This is a complex task, which requires expert knowledge, much experience and fast reaction. There are a large number of factors related to a traffic state as well as a large number of possible control measures that need to be considered during the decision making process. The identification of suitable control measures for a given non-recurrent traffic congestion can be tough even for experienced operators. Therefore, simulation models are used in many cases. However, simulating different traffic scenarios for a number of control measures in a complicated situation is very time-consuming. In this paper we propose an intelligent traffic control decision support system (ITC-DSS) to assist the human operator of the traffic control centre to manage online the current traffic state. The proposed system combines three soft-computing approaches, namely fuzzy logic, neural network, and genetic algorithm. These approaches form a fuzzy-neural network tool with self-organization algorithm for initializing the membership functions, a GA algorithm for identifying fuzzy rules, and the back-propagation neural network algorithm for fine tuning the system parameters. The proposed system has been tested for a case-study of a small section of the ring-road around Riyadh city. The results obtained for the case study are promising and show that the proposed approach can provide an effective support for online traffic control.

1 Introduction

Cities around the world face serious traffic congestion problems as the number of vehicles and the need for transportation grow. Traffic congestions do not only cause considerable costs due to unproductive time losses; but they also increase the probability of accidents as well as have negative impacts on the environment (air pollution, lost fuel) and on the quality of life (health problems, noise, stress) [1]. Therefore, the traffic management and control have been a major problem in developing as well as developed countries. Governments have been spending hefty amounts to develop the traffic control centres using different methodologies and advanced information technology.

Contemporary traffic control centres are connected to devices such as detectors, weather sensors, and cameras to record data related to the traffic state on-line, e.g. speed, flow, demand, environmental conditions, etc. Moreover, the control centres use

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 688–701, 2007.
© Springer-Verlag Berlin Heidelberg 2007

advanced dynamic control devices such as ramp metering, dynamic route information panels (DRIPs) and/or variable message signs (VMSs). Figure 1 shows a typical infrastructure for real-time traffic control that can be found in different cities [2].

When non-recurrent congestion happens, the operator of the traffic control centre has to assess the severity of the situation, predict the most probable evolution of the state of the network, and select the most appropriate control measures quickly [3]. This is a complex task, which requires expert knowledge and much experience, which can often only be obtained after an extensive training. There are a large number of factors related to a traffic state and a large number of possible control measures that need to be considered during the decision making process. The identification of suitable control measures for a given non-recurrent traffic congestion can be tough even for experienced operators [4]. Therefore, an advanced traffic control system that integrates the traffic state data with traffic monitoring and control software to help operators in decisions making is needed. Road traffic simulation models are used in many cases. However, simulating different traffic scenarios for a number of control measures in a complicated situation can be very time-consuming [4].

In this paper we propose an intelligent traffic control decision support system (ITC-DSS) to assist the human operator of the traffic control centre to manage online the traffic on highways and urban ring roads. The inputs of the system are the current traffic state and the available control measures, and the output is a ranked list of all combinations of control measures. The proposed ITC-DSS is based on fuzzy logic approach in combination with other soft-computing approaches; neural network, and genetic algorithm. These approaches within the ITC-DSS provide the initialization and self-organization of the membership functions for fuzzy input variables,

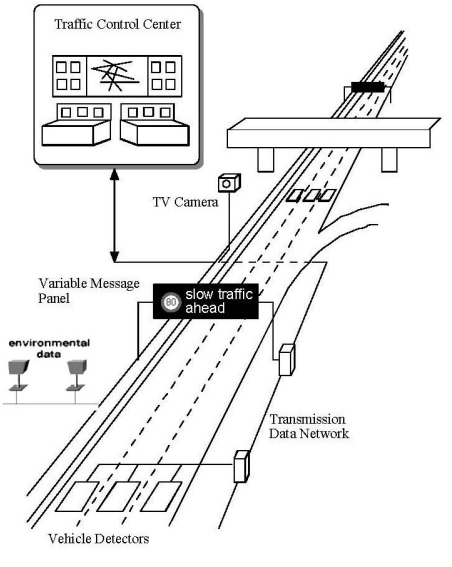


Fig. 1. Typical information infrastructure for real-time traffic control [2]

identification of fuzzy rules, and fine tuning of the system parameters. We have tested the system for a case-study of a small section of the ring-road around Riyadh city in Saudi-Arabia.

The paper is organized as follows. The next section reviews the related work published in the literature. Section 3 describes the proposed ITC-DSS, its design and functions. This is followed by the application of the developed ITC-DSS for a case study in Section 4. The conclusions and future works of this paper are given in Section 5.

2 Related Work

Several authors have described decision support systems for traffic control using different intelligent approaches [2-11]. Some of these applications used the fuzzy logic technique in their decision process [3, 4, 11] while others used neural networks [8, 10]. The TRYS system described in [2, 7] is a an agent-based environment for building intelligent traffic management systems applications for urban, interurban and mixed traffic areas. TRYS system is based on knowledge frames, and some of these frames use fuzzy logic. Other knowledge-based real-time decision support systems for road traffic management are described in [5, 6]. Hegyi et al [3] have presented a fuzzy logic based traffic control system for efficiently managing non-recurrent congestions, which later has been extended by De Schutter et al in [4] as a multi-agent traffic control system. The presented fuzzy traffic control system is part of a larger traffic support system which uses a case base and fuzzy logic to generate a ranked listing of combinations of traffic control measures and their estimated performance for a given traffic situation. Since the proposed system does not use any knowledge form experts or heuristic rules and it is only based on a case based reasoning system, the quality of its result depends basically on the quality of the case base. Almejalli et al [11] have proposed a fuzzy based decision support system to improve long-term traffic management policies. The proposed system assists traffic decision makers by providing recommendations on whether they should, for example, give priority to increasing the number of traffic policemen on duty, increasing fines, or installing better/more safety devices.

Ramp controlling or ramp metering is a traffic control technique used to control the traffic inflow into a freeway by limiting the number of vehicles entering the freeway. Usually, the main goal of the ramp metering system is to avoid congestion and reduce vehicle's total travel time. For solving freeway ramp-metering control problems, Wei [8] has developed artificial neural network models. Inputs to neural network models are traffic states in each time period on the freeway segments while outputs are the desired metering rate at each entrance ramp. Also Zhang et al [10] have used the neural network technique for ramp control, while Bogenberger et al [9] have applied adaptive fuzzy logic system for ramp controlling problem. The adaptive fuzzy logic algorithm has been used to determine the traffic responsive metering rate.

Fuzzy Neuron Network has been employed in the traffic management field in several papers. For example, Henry et al [12] have developed a neuro-fuzzy control method for controlling of traffic lights of an intersection. The system showed good results for simple and medium-complexity intersections but poor performance on a

complex intersection. Another fuzzy neuron network system has been proposed in [13] to the analysis and prediction of traffic flow. The system has been fully trained and subsequently used for short-term traffic flow prediction. The prediction results are shown to be promising.

In this paper we are extending the idea of the fuzzy decision support system for traffic control presented in [3, 4] to using a similar structure of the fuzzy neural network discussed in [13] for self-organization and initialization of the fuzzy sets and membership functions. We also incorporate a GA-based offline learning algorithm to generate the fuzzy rules to this adaptive fuzzy neural network. The proposed decision support system is detailed in the next section.

3 The Proposed Decision Support System

3.1 Overall ITC-DSS Framework

As mentioned earlier a large number of factors which determines the current traffic state need to be considered in the decision making process. These input factors are usually measured by the on-line monitoring system using sensors, detectors and cameras (Alternatively, the traffic state can be forecasted by a traffic flow model). They include traffic densities, average speeds, traffic demand, etc. Similarly, there are many possible control measures that can be employed to control the road network depending upon the nature of traffic problems and available road control facilities. The proposed system receives online the current traffic values (from monitoring system) and

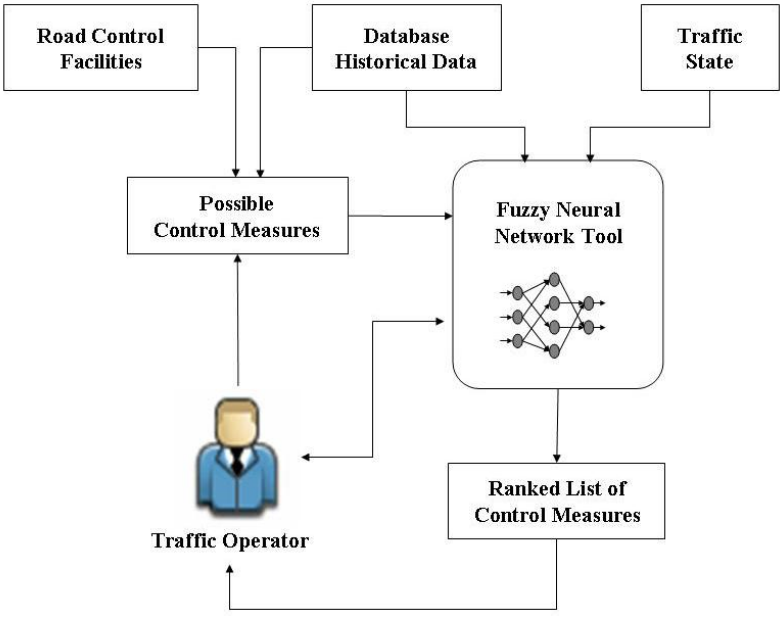


Fig. 2. Structure of ITC-DSS

based on that it assists the human operator of the traffic control centre to manage online the traffic network. The overall structure of the proposed system is depicted in Figure 2.

In general, the proposed ITC-DSS works as follows. Let *S* be the set of all possible control measures, which can be used to control the considered road network. *S* is created for a given road network off-line using the available road control facilities, traffic operator's experience and the historical data. Each control measure is represented by $c_i \in S$, where *i* varies from 1 to *n*, and *n* is the number of possible measures. Given a current traffic state from the on-line monitoring system and set of all possible control measures (*S*) for the given road network the ITC-DSS employs a pre-trained fuzzy-neural network tool (FNN_Tool) (see detail in the next section) to predict the evaluation of each c_i for the current traffic state over a range of performance criteria such as queue lengths, total travel times, fuel consumption, etc. The aggregated performance of each c_i is calculated using a weighted sum [4] , which is defined as:

$$P_{C_i} = \sum_{k=1}^{N} w_k E_{k,Ci} \tag{1}$$

where P_{C_i} is the aggregated performance of control measure C_i for the given traffic state ; $E_{k,Ci}$ is the evaluation of control measure C_i over performance criterion *k* for the given traffic state; w_k is the weight of the performance criterion *k*; and *N* is the number of considered performance criteria. These weights (w_k) are usually selected by the operators based on the current traffic management policies and other considerations. Using the aggregated performance P_{C_i} the system then provides the operator with a ranked list of the control measures in real-time. The following pseudo-code summarises the main functions of the proposed ITC-DSS:

```
Set Traffic_State_Input = Current traffic State;
Identify S = { c1,c2,c3,....,cn};
For ( i=1; i<=n; i++) do
     {
     Set Control_Input = ci  ;
     Run FNN_Tool(Traffic_State_Input, Control_Input);
     Obtain evaluations over all performance criteria
     Calculate aggregated performance measure
     }
Rank ci ∈ S ;
Show ranked list;
```

3.2 Fuzzy Neural Network Tool (FNN_Tool) Structure

In principle a simple neural network can be used as a decision support tool within the proposed model. It can give an accurate output provided it is trained with all possible cases. However, given the high-dimensionality of the prediction problem addressed

here, training a neural network on all possible traffic cases is impossible. For example, if the conditions in a network are described by the period of day, densities of its links, traffic demands on the network boundaries, control measures that have been applied, and the incident status; then the description of the conditions on a 25 link network will yield approximately 10^{24} cases [14]. Clearly, it is unfeasible to consider such a number of traffic cases in the training process. Therefore, fuzzy neural network is used to address this problem.

Fuzzy neural networks are hybrid intelligent systems, which combine the advantages of both neural networks and fuzzy logic. The neural fuzzy system is a fuzzy system that uses the learning ability of the neural networks to determine its parameters (fuzzy sets, fuzzy memberships and fuzzy rules) by processing data. In other words, neural fuzzy systems aim at providing fuzzy systems self-adaptive capability with the automatic tuning method of neural networks. In a neural fuzzy system, the neural network helps the fuzzy system to elicit membership function, map fuzzy sets to fuzzy rules, and implement defuzzification [15, 16].

The structure of the neural fuzzy network tool (FNN_Tool) used in our ITC-DSS is similar to the structure proposed in [13]. It is a five-layer structure, as shown in Figure 3, where each layer performs an operation for building the fuzzy system. The inputs of our FNN_Tool are the current traffic state which is characterized by input factors e.g., densities, average speeds, traffic demand; and all possible control measures. The outputs of the FNN_Tool are the evaluation of those control measures for the current traffic state over a number of performance criteria, such as, queue lengths,

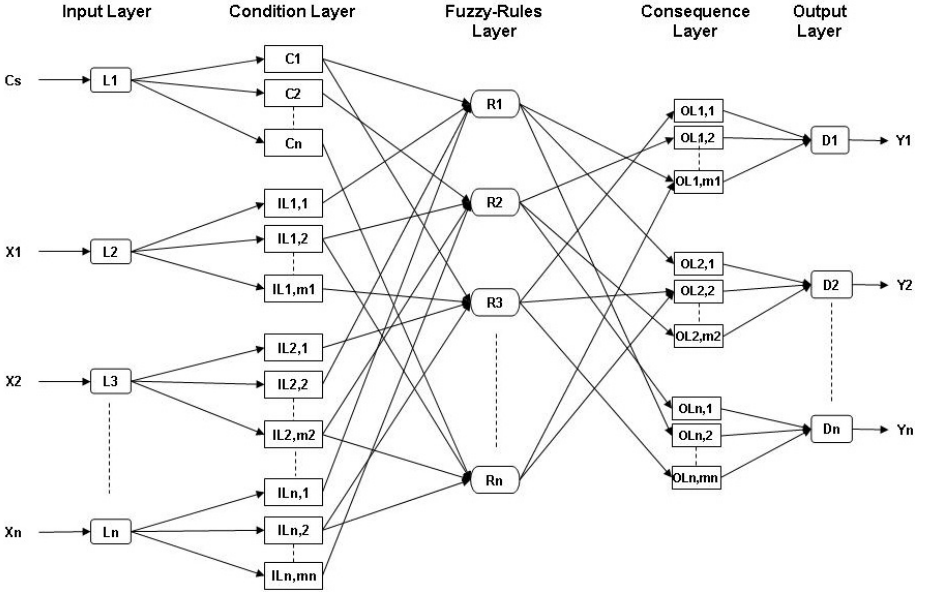


Fig. 3. Structure of the fuzzy neural tool (FNN_Tool)

total travel times, waiting times. The process of each layer is described below (see Figure 3):

Layer 1: is the input layer. Neurons in this layer represent input linguistic variable such as "speed", "density", and "control measure" and directly transmit non-fuzzy input values to the next layer. In our case, the neurons inputs represent the *Traffic_input* of the current traffic state, which are represented as vectors $X^T = [x_1, x_2, ..., x_n]$, and the *Control_input* of the road network, which is represented as C_S. The link weight, w_i^1, between this layer and the next layer is 1. The input and the output of this layer are given as follows:

$$o_i^{(1)} = i_i^{(1)} \tag{2}$$

where $i_i^{(1)}$ is the input and $o_i^{(1)}$ is the output of input neuron i in layer 1.

Layer 2: is the fuzzification layer, which defines the fuzzy sets and membership for each of the input factors. Neurons in this layer acts as a membership function and represents the terms of the respective linguistic variable, such as "low", "high", and "measure 1". In our model the neurons of this layer are modeled as a common bell-shaped membership function [17], so the input $i_{i,j}^{(2)}$ and the output $o_{i,j}^{(2)}$ of fuzzification neuron i in the layer 2 are given as follows:

$$o_{i,j}^{(2)} = e^{-\frac{(i_{i,j}^{(2)} - m_{i,j})^2}{\sigma_{i,j}}} \tag{3}$$

where $m_{i,j}$ and $\sigma_{i,j}$ are the centres and the widths of the membership function for the input-label neuron $LI_{i,j}$ respectively.

Layer 3: is the fuzzy rule layer, which defines all possible fuzzy rules to specify qualitatively how the output parameter is determined for various instances of the input parameters. Each neuron in this layer represents a fuzzy rule, for example, "if the time is morning, average speed is high, density is low and control measure is "measure 1", then total travel time is medium and waiting times is low." Different approaches are used to identify the fuzzy rules in the fuzzy neural networks, such as using linguistic information from experts to identify fuzzy rules [18, 19], using unsupervised learning algorithms [20, 21], and using supervised leaning algorithm (particularly the Back-propagation technique) [22]. In this paper we have proposed a GA based learning algorithm to identify fuzzy rules (see Section 3.3 for detail). The input and the output of a rule neuron in the layer 3 are given as follows:

$$y_i^{(3)} = \min(x_{ki}^{(3)}) \tag{4}$$

where $x_{ki}^{(3)}$ are the inputs, and $y_i^{(3)}$ is the output of fuzzy rule i in the layer 3

Layer 4: is the consequence layer (or the output membership layer). Neurons in the consequence layer represent fuzzy sets (such as "low" and "high") used in the consequent

part of the fuzzy rules. The input and the output of a consequence neuron in the layer 4 are given as follows:

$$y_i^{(4)} = \min \left(1, \sum_l x_{l,i}^{(4)}\right) \tag{5}$$

where $x_{l,i}^{(4)}$ is the input (the output of neuron l in the fuzzy rule layer), and $y_i^{(4)}$ is the output of membership neuron i in the layer 4.

Layer 5: is the output layer (or the defuzzification layer). Each neuron in the output layer represents a single output variable such as "average speed" and "travel time". The input and the output of an output neuron in layer 5 are given as follows:

$$x_i^{(5)} = \sum (a_{ci} / b_{ci}) \times y_i^{(4)} \tag{6}$$

$$y_i^{(5)} = \frac{x_i^{(5)}}{\sum_{i=1} (y_i^{(4)} / b_{ci})} \tag{7}$$

where $x_i^{(5)}$ is the input and $y_i^{(5)}$ is the output neuron i in layer 5, a_{ci} and b_{ci} are the centre and the width of the fuzzy set respectively.

3.3 Learning Process of ITC-DSS

We propose to use three stages of learning for our ITC-DSS. First stage is initializing the membership functions of both input and output variables by determining their centres and widths. To perform this stage, we have employed a self-organizing algorithm [23] as in other works [13, 20, 24]. A proposed GA based learning algorithm is performed in the second stage to identify the fuzzy rules that are supported by the set of training data. In the last stage, the derived structure and parameters are fine tuned by using the back-propagation learning algorithm [17]. In the following section, a brief description of the proposed GA based learning algorithm is presented.

3.4 GA-Based Learning Algorithm

Several authors have proposed a genetic algorithm for optimize fuzzy neural parameters adjusting the control points of membership functions or for tuning the weightings [25, 26]. In this paper we use a genetic algorithm to generate the fuzzy rules needed for constructing the fuzzy neural network. To explain how we have designed a GA for generating fuzzy rules, consider a simple example of FNN_Tool with two input linguistic variables x_1 and x_2 , and one output linguistic variable *y* as shown in Figure 4.

After performing the self-organization learning algorithm, each linguistic variable has a number of fuzzy sets, say we have three fuzzy sets {low (L), medium (M), high (H)}. The proposed GA learning algorithm considers all possible rules. In our simple example there are a total of twenty seven possible rules. In fact these rules are made of nine possible antecedents (preconditions)., These antecedents of fuzzy rules are

represented by neurons R_1 … R_9 of the Fuzzy-Rules Layer in Figure 4. Each antecedent has links with three possible decision fuzzy sets (neurons in Consequence Layer: L, M and H). For example, the three possible fuzzy rules associated with neuron R_1 are:

If x_1 is L and x_2 is L, then y is L.
If x_1 is L and x_2 is L, then y is M.
If x_1 is L and x_2 is L, then y is H.

A number of decisions must be made in order to implement the GA for generating appropriate fuzzy rules. There are problem specific decisions which are concerned with the search space (and thus the representation) and the form of the fitness function.

The encoding of the problem using an appropriate representation is a crucial aspect of the implementation of the GA technique. The encoding used to represent chromosomes (solutions) defines the size and the structure of the search space. Here we propose integer strings as chromosomes to represent candidate solutions of the problem. The string is given by $t_1,t_2,...,t_i,...,t_N$, where t_i is an integer $0 \le t_i \le M$ which indicates the link of neuron R_i (i.e. neurons in Fuzzy-Rules Layer) with output neurons (i.e., neurons in Consequence Layer). N is the number of neurons in the Fuzzy-Rules Layer and M is the number of neurons in the Consequence Layer. For our example the chromosome has nine integers, and $0 \le t_i \le 3$. $t_i = 0$ indicates there is no link of R_i with output neuron; $t_i = 1$ indicates that there is a link with 'L' neuron in Consequence Layer and so on.

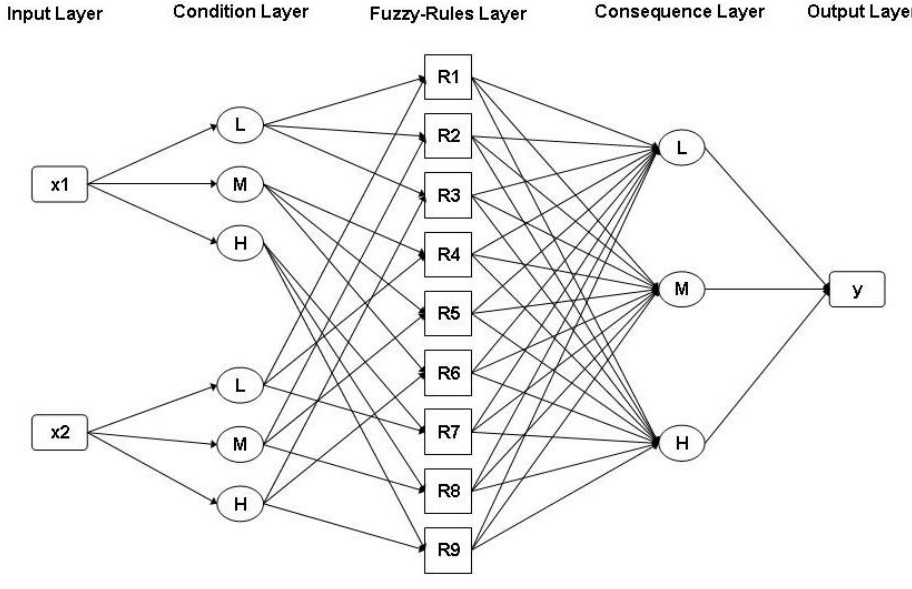


Fig. 4. FNN_Tool with two inputs variable and one output variable with all possible fuzzy rules

The goodness of every chromosome is evaluated by using a fitness function. We use a set of training data to calculate the fitness of each chromosome based on the following fitness function:

$$\text{Fitness} = \frac{1}{1 + |e|} \tag{8}$$

where e is the difference (error) between the actual output of the fuzzy-neural network and the desired output. The GA aims to maximize the fitness function (8) to minimize the error value (e).

Based on our previous experience with GA and a number of experiments we have selected GA operators and their parameters to be used for this application. The GA operators used are steady state replacement approach, tournament selection, standard two-point crossover and standard random mutation. The steady state approach directly inserts a new solution into the population pool replacing a less fit solution. The tournament selection method picks a subset of solutions at random from the population to form a tournament selection pool, from which two solutions are selected with probability based upon the fitness values of the solutions. The two-point crossover operator splits the selected solutions at two randomly chosen positions and exchanges the centre sections with probability a crossover probability. The mutation operator changes the integer at each position in the solution within the allowed range with a defined mutation probability. The elitist approach, which ensures that the best solution in the population pool is always retained, has been applied. The initial population of chromosomes is created randomly. The stopping criterion for a GA run is to achieve the pre-specified error level (e).

When the GA learning process has completed (i.e. when pre-specified error level is achieved) after running the GA over a large number of runs, we choose the best GA chromosome. This best chromosome is decoded to get the structure of the FNN_Tool by keeping only the links that are indicated by the chromosome. The GA approach can take a lengthy computation time for the optimization process. This computation time is however not considered problematic for off-line training.

4 Case Study

In order to evaluate and test the proposed ITC-DSS discussed in the previous section, we have derived a test case study using a small section of the ring-road around Riyadh, the capital city of Saudi-Arabia. The selected section is one of the busiest parts of the Riyadh ring-road, because it is used mostly for traffic going to the city centre as shown in Figure 5. This section includes 10-km of the main road with three lanes in each way and a service road with a limited capacity. The service road separates from the main road at point *B* and runs parallel to the main road and gives access to Mather Street and then joins the main road again at point *C* (see Figure 5). *A*, *B*, and *C* are join points between service roads and the main road and they are controlled by ramp metering devices. Before point *B* there is a DRIP that can display queue information, or give some alternative routes to drivers. In this case study, we only consider traffic going from the south to the north.

Since the aim of this stage is assessing the technical feasibility of the proposed system, only a limited number of inputs, control measures, and training data have been

considered. However, the increase in the number of inputs, possible control measures, and training data should not affect the validity of the proposed system. We have considered the following variables in our case study:

Two traffic factors to represent the current traffic state:

- Traffic flow (vehicle/hour): on both the main road and the service road.
- Traffic demand (number of vehicles): for both the main road and the service road.

Five traffic control measures:

- C1: metering the on-ramp at point *A*, and using DRIP.
- C2: metering the off-ramp at point *B*, and using DRIP.
- C3: metering the on-ramp at points *A* and *C*, and off-ramp at points *B*.
- C4: using DRIP to display the queue information.
- C5: doing nothing.

We use three evaluation criteria for calculating the performance of the measures:

- Total travel time (TTT) (hours).
- Total fuel consumption (TFC) (liters)
- Average queue length (AQL) (number of vehicles).

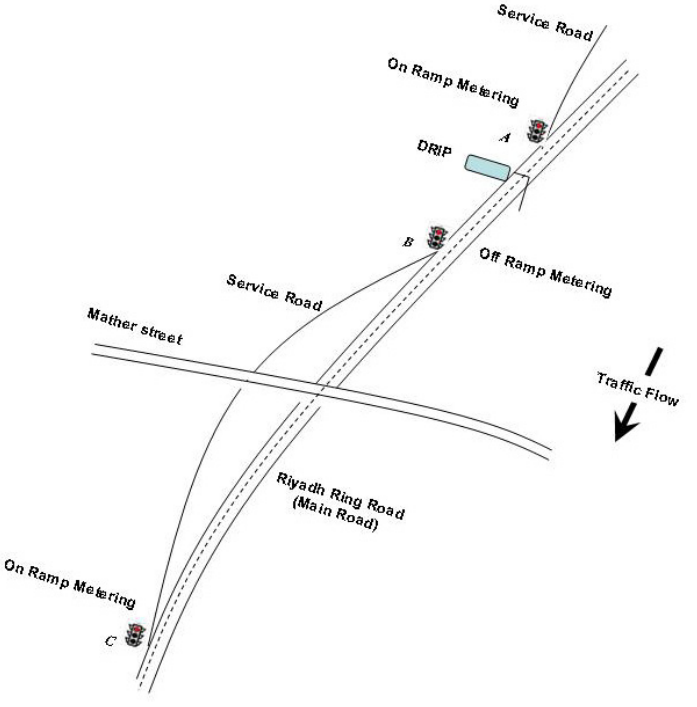


Fig. 5. The sub network considered in the prototype

The data needed for the training process has been generated using a traffic simulation model (more specifically, the METANET macroscopic flow model [27]). All our variables have been considered and simulated for the period from 4pm to 8pm.

Table 1. The performance evaluation of the control measures C1-C5 on a selected traffic state

Traffic State :	Traffic Flow : Main Road = 2600 Service Road =1000 Traffic Demand: Main Road = 1780 Service Road = 900					
Control Measures	Riyadh –Traffic Simulation Model			ITC-DSS		
	TTT	TFC	AQL	TTT	TFC	AQL
C1	1243	3589	15	1240	3581	15
C2	987	2989	19	984	2988	19
C3	1056	3345	22	1050	3345	21
C4	890	2879	15	888	2872	15
C5	1267	3108	17	1266	3103	16

In order to test the performance of ITC-DSS, we have made a comparison between ITC-DSS and the traffic simulation model. The results obtained for the performance of the five control measures by ITC-DSS and the simulation model on a selected traffic state are shown in Table 1. We found that the results obtained by ITC-DSS are very close to the results given by the traffic simulation model, which confirms the validity of the proposed model in obtaining results. However, the time ITC-DSS needs to calculate the performance of a given control measure is much less than the time needed by the simulation model. Since ITC-DSS has been developed using trained FNN_Tool, the main advantage of ITC-DSS is its speed. Also, ITC-DSS allows the user to evaluate a set of control measures in one process instead of giving one by one.

5 Conclusion and Future Work

This paper has described an online decision support system for a traffic control centre. The proposed system uses a decision support tool with FL, NN and GA techniques to assist the human operator of the traffic control center to manage the current traffic state. When a non-recurrent congestion happens, the operator can use the proposed system to assess approximate performance of several control measures online. For constructing and training the fuzzy neural network tool used in the proposed system we have performed three different algorithms: self-organization algorithm for initializing the membership functions, a GA algorithm for identifying fuzzy rules, and the back-propagation algorithm for fine tuning the system parameters. In order to evaluate and test the proposed system, a case study of a section of the ring-road around Riyadh is presented and discussed. The results of the proposed system clearly demonstrate its merits and capabilities in terms of processing speed and flexibility.

This research investigation has demonstrated the technical feasibility of the proposed system with a traffic network and limited inputs (traffic variables and control

measures). In the next stage, we will assess the performance of the proposed system for a larger traffic network with more traffic variables and control measures.

Finally, it is noted that the proposed ITC-DSS can provide an effective tool to assist traffic control operators in real-time decision making for a traffic network with a large number of traffic variables and control measures. In future we aim to extend the proposed system for on-line learning for a complex traffic network.

References

1. FHWA. *FHWA Administrator Testifies That Growing Traffic Congestion Threatens Nation's Economy, Quality of Life*. 2002 [cited 2006 15 October 2006]; Available from: http://www.fhwa.dot.gov/pressroom/fhwa0220.htm.
2. Molina, M., J. Hern A, and J.E. Cuena, *A structure of problem-solving methods for real-time decision support in traffic control.* International Journal of Human-Computer Studies, 1998. **49**(4): p. 577.
3. Hegyi, A., et al., *A fuzzy decision support system for traffic control centers.* Intelligent Transportation Systems, 2001. Proceedings. 2001 IEEE, 2001: p. 358-363.
4. De Schutter, B., et al., *A multi-agent case-based traffic control scenario evaluation system.* Intelligent Transportation Systems, 2003. Proceedings. 2003 IEEE, 2003. **1**: p. 678-683.
5. Ritchie, S.G., *A knowledge-based decision support architecture for advanced traffic management.* Transportation Research Part A: General, 1990. **24**(1): p. 27.
6. Zhang, H. and S.G. Ritchie, *Real-Time Decision-Support System for Freeway Management and Control.* Journal of Computing in Civil Engineering, 1994. **8**(1): p. 35-51.
7. Cuena, J., J. Hernandez, and M. Molina, *Knowledge-based models for adaptive traffic management systems.* Transportation Research Part C: Emerging Technologies, 1995. **3**(5): p. 311-337.
8. Wei, C.H., *Analysis of artificial neural network models for freeway ramp metering control.* Artificial Intelligence in Engineering, 2001. **15**(3): p. 241-252.
9. Bogenberger, K. and H. Keller, *An evolutionary fuzzy system for coordinated and traffic responsive ramp metering.* System Sciences, 2001. Proceedings of the 34th Annual Hawaii International Conference on, 2001: p. 10.
10. Zhang, H.M., S.G. Ritchie, and R. Jayakrishnan, *Coordinated traffic-responsive ramp control via nonlinear state feedback.* Transportation Research Part C: Emerging Technologies, 2001. **9**(5): p. 337-352.
11. Almejalli, K., K. Dahal, and A. Hossain, *Road Traffic Decision Support System* to apprear in the proceedings of Software, Knowledge Information Management and Applications (SKIMA 2006), 2006.
12. Henry, J.J., J.L. Farges, and J.L. Gallego, *Neuro-fuzzy techniques for traffic control.* Control Engineering Practice, 1998. **6**(6): p. 755-761.
13. Quek, C., M. Pasquier, and B.B.S. Lim, *POP-TRAFFIC: a novel fuzzy neural approach to road traffic analysis and prediction.* IEEE Transactions on Intelligent Transportation Systems, 2006. **7**(2): p. 133- 146.
14. Hoogendoorn, S.P., H. Schuurman, and B. De Schutter, *Real-time traffic management scenario evaluation.* Proceedings of the 10th IFAC Symposium on Control in Transportation Systems (CTS 2003), 2003: p. 343–348.
15. Lin, T., *Neural Fuzzy Control Systems With Structure and Parameter Learning*. 1994, Singapore: World Scientific
16. Tay, J.H. and X. Zhang, *Neural Fuzzy Modeling of Anaerobic Biological Wastewater Treatment Systems.* Journal of Environmental Engineering, 2006. **125**(12): p. 1149-1159.

17. Rumelhart, D.E., G.E. Hinton, and R.J. Williams, *Learning internal representations by error propagation, Parallel distributed processing: explorations in the microstructure of cognition, vol. 1: foundations*. 1986, MIT Press, Cambridge, MA.
18. Ronald, R.Y., *Implementing fuzzy logic controllers using a neural network framework.* Fuzzy Sets and Systems, 1992. **48**: p. 53-64.
19. Krause, B., et al., *A neuro-fuzzy adaptive control strategy for refuse incineration plants.* Fuzzy Sets and Systems, 1994. **63**(3): p. 329-338.
20. Lin, C.T. and C.S.G. Lee, *Neural-Network-Based Fuzzy Logic Control and Decision System.* IEEE Transactions on Computers, 1991. **40**(12): p. 1320-1336.
21. Quek, C. and R.W. Zhou, *The POP learning algorithms: reducing work in identifying fuzzy rules.* Neural Netw, 2001. **14**(10): p. 1431-45.
22. Lee, M., S.Y. Lee, and C.H. Park, *A New Neuro-Fuzzy Identification Model of Nonlinear Dynamic Systems.* International Journal of Approximate Reasoning, 1994. **10**(1): p. 29-44.
23. Kohonen, T., *Self-Organization and Associative-Memories.* 1984, Springer-Verlag: New York.
24. Werbos, P.J., *Neurocontrol and fuzzy logic: connections and designs.* International Journal of Approximate Reasoning, 1992. **6**(2): p. 185-219.
25. Wang, W.Y., et al., *GA-based learning of BMF fuzzy-neural network.* Fuzzy Systems, 2002. FUZZ-IEEE'02. Proceedings of the 2002 IEEE International Conference on, 2002. **2**.
26. Wang, W.Y., C.Y. Cheng, and Y.G. Leu, *An online GA-based output-feedback direct adaptive fuzzy-neural controller for uncertain nonlinear systems.* Systems, Man and Cybernetics, Part B, IEEE Transactions on, 2004. **34**(1): p. 334-345.
27. Technical University of Crete, Dynamic Systems and Simulation Laboratory, and A. Messmer, *METANET – A simulation program for motorway networks*. July 2000.

An Ant-Based Heuristic for the Railway Traveling Salesman Problem

Petrica C. Pop[1], Camelia M. Pintea[2], and Corina Pop Sitar[3]

[1] Department of Mathematics and Computer Science,
Faculty of Sciences, North University of Baia Mare, Romania
pop_petrica@yahoo.com
[2] Faculty of Mathematics and Computer Science, Babes-Bolyai University,
Cluj-Napoca, Romania
cmpintea@cs.ubbcluj.ro
[3] Faculty of Economics, Babes-Bolyai University of Cluj-Napoca, Romania
sitar@conseco.ro

Abstract. We consider the *Railway Traveling Salesman Problem*, denoted *RTSP*, in which a salesman using the railway network wishes to visit a certain number of cities to carry out his/her business, starting and ending at the same city, and having the goal to minimize the overall time of the journey. The *RTSP* is NP-hard and it is related to the *Generalized Traveling Salesman Problem*. In this paper we present an effective meta-heuristic based on ant colony optimization (*ACO*) for solving the *RTSP*. Computational results are reported for real-world and synthetic data. The results obtained demonstrate the superiority of the proposed algorithm in comparison with the existing method.

1 Introduction

We consider a problem of central interest in railway optimization. We assume that we are given a set of stations, a timetable regarding trains connecting these stations, an initial station, a subset B of the stations and a starting time. A salesman wants to travel from the initial station, starting not earlier than the designated time, to every station in B and finally returns back to the initial station, subject to the constrained that he/she spends the necessary amount of time in each station of B to carry out his/her business. The goal is to find a set of train connections such that the overall time of the journey is minimized. This problem was introduced in [7] and called the *Railway Traveling Salesman Problem* (*RTSP*).

The *RTSP* is related to the *Generalized Traveling Salesman Problem* (*GTSP*) [5,6]. In that problem, a weighted complete directed graph is given whose nodes are partitioned into a given number of clusters. The goal is to find a minimum-weighted tour containing *exactly* one node from each cluster. By virtue of *TSP*, the *GTSP* is an NP-hard problem. It can be easily seen that the *TSP* is polynomially time reducible to the *RTSP* as well, for each pair of cities, for which there exists a connection, consider a train leaving from the first one to the second

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 702–711, 2007.
© Springer-Verlag Berlin Heidelberg 2007

with travel time equal to the cost of the corresponding connection in *TSP*. This simple reduction sets the *RTSP* in the class of NP-hard problems.

Hadjicharalambous *et al.* in [7] presented an integer programming formulation for the *RTSP* and based on a size reduction of the graph, they solved the proposed IP model using CPLEX.

Due to the complexity of the *RTSP*, the problem cannot be solved within polynomially bounded computation times. Nevertheless, sub-optimal solutions are sometimes easy to find. Consequently, we will focus on heuristic algorithms that can find near optimal solutions within reasonable running time. Heuristic algorithms are typically among the best strategies in terms of efficiency and solution quality for problems of realistic size and complexity.

In contrast to individual heuristic algorithms that are designed to solve a specific problem, meta-heuristics are strategic problem solving frameworks that can be adapted to solve a wide variety of problems. Meta-heuristic algorithms are widely recognized as one of the most practical approaches for combinatorial optimization problems.

Meta-heuristics inspired from nature represent a powerful and robust approach to solve NP-difficult problems. Biology studies emphasize the remarkable solutions that many species managed to develop after millions of years of evolution. Self-organization and indirect interactions between individuals make possible the identification of intelligent solutions to complex problems. These indirect interactions occur when one individual modifies the environment and other individuals respond to the change at a later time. This process refers to the idea of *stigmergy*, [1]. An example of this is ant colonies: every single ant just seems to walk independently, however the colony itself is organized very well. Ants' foraging behavior shows that travel between nests and sources of food is highly optimized.

In this paper we present an effective meta-heuristic based on ant colony optimization (*ACO*) for solving the *RTSP*. Computational results are reported for real-world and synthetic data and the results obtained demonstrate the superiority of the proposed algorithm in comparison with the existing ones.

2 Ant Colony Optimization

The *Ant Colony Optimization*(*ACO*) is a bio-inspired meta-heuristic composed of different algorithms in which several cooperative agent populations try to simulate real ant behavior. Dorigo, Di Caro and Gambardella [2,3] proposed the *Ant Colony Optimization*.

The insects behavior is replicated to search the space. While walking between their ant nest and the food source, ants deposit a substance called *pheromone*. In the near future, every ant can direct its search (and therefore direct the search of the whole colony as a group) according to the amounts of this chemical substance on the ground. When ants arrive to a path intersection, they need to choose the path to follow. Each ant selects it applying a probabilistic decision biased by the amount of pheromone: stronger *pheromone trails* are preferred.

The most promising paths receive a greater pheromone trail after some time. The continuous movement of every ant in the colony causes the best paths (the shortest ones) to have the largest amounts of pheromone the shorter the path the faster ants go through it and consequently more ants walk over the path and more pheromone is left behind. On the other hand, longer paths are progressively abandoned. Therefore, the pheromone that was deposited finally evaporates. At the end, the best path (the minimum length one) is found between the ant nest and the food source.

An *ACO* algorithm is essentially a system based on agents, artificial ants, which simulate the natural behavior of ants including mechanisms of cooperation and adaptation. In [2], the use of this kind of system as a new meta-heuristic was proposed in order to solve combinatorial optimization problems.

Ant Colony Optimization meta-heuristic has been shown to be both robust and versatile in the sense that it has been successfully applied to a range of different combinatorial optimization problems, [3,4] .

ACO algorithms are based on the following main ideas:

- Each path followed by an ant is associated with a candidate solution for a given problem.
- When an ant follows a path, the amount of pheromone deposited on that path is proportional to the quality of the corresponding candidate solution for the target problem.
- When an ant has to choose between two or more paths, the path(s) with a larger amount of pheromone has(have) a greater probability of being chosen by ants. As a result, ants eventually converge to a short path which hopefully represents the optimum or a near-optimum solution for the target problem.

3 The Railway Traveling Salesman Problem

In this section, we describe the input of an *RTSP* instance and the time-expended graph on which the problem will be defined. In the following we assume timetable information in a railway system, but the modeling and the solution approach can be applied to any other public transportation system provided that it has some characteristics.

A *timetable* consists of data concerning: stations, trains, connecting stations, departure and arrival times of trains at stations. More formally, we are given a set of trains Z, a set of stations S, and a set of *elementary connections* C whose elements are 5-tuples of the form $(z, \sigma_1, \sigma_2, t_d, t_a)$. Such a tuple (elementary connection) is interpreted as follows: the train z leaves the station σ_1 at time t_d and the next stop of the train z is station σ_2 at time t_a. The departure and arrival times t_d and t_a are integers in the interval $T_{day} = [0, 1439]$ representing time in minutes after midnight.

Given two time values t and t' with $t \leq t'$, their *cycle-difference*(t, t') is the smallest nonnegative integer l such that $l = t' - t \pmod{1440}$.

We are also given a starting station $\sigma_0 \in S$, a time value $t_0 \in T_{day}$ denoting the earliest possible departure time from σ_0 and a set of stations $B \subseteq S - \sigma_0$,

which represents the set of stations (cities) that the salesman has to visit. A function $f_B : B \rightarrow T_{day}$ is used to model the time that the salesman has to spend at each city $b \in B$, i.e., the salesman must stay in station $b \in B$ at least $f_B(b)$ minutes.

We naturally assume that the salesman does not travel continuously and that if he/she arrives too late in some station, then he/she has to rest and spend the night there. Moreover, the salesman's business for the next day may not require taking the first possible train from that station. Consequently, we assume that the salesman never uses a train that leaves too late in the night or too early in the morning.

The *RTSP* can be modeled as a graph theory problem, making use of the so-called *time-expanded* digraph introduced in [9]. Such a graph $G = (V, E)$ is constructed using the provided timetable information as follows. The nodes of the graph are partitioned into clusters representing the set of nodes associated to a station in the expanded graph. There is a node for every time event (departure or arrival) at a station, and there are three types of edges. For each elementary connection $(z, \sigma_1, \sigma_2, t_a, t_d)$ in the timetable, there is a *train-edge* in the graph connecting a departure node, belonging to the station σ_1 and associated with the time t_d, with an arrival node, belonging to the station σ_2 and associated with the time t_a. For each station $\sigma \in S$, all departure nodes belonging to σ are ordered according to their time values. Let $v_1, ..., v_k$ be the nodes of σ in that order. Then, there is a set of *stay-edges*, denoted by $Stay(\sigma)$, (v_i, v_{i+1}), $1 \leq i \leq k-1$ and (v_k, v_1) connecting the time events within a station and representing waiting within a station. Additionally, for each arrival node in a station there is an *arrival-edge* to the immediately next (w.r.t. their time values) departure node of the same station. The cost of an edge (u, v) is *cycle-difference*(t_u, t_v), where t_u and t_v are the time values associated with u and v.

In order to model the *RTSP*, we introduce the following modifications to the time-expanded digraph:

- we do not include any elementary connections that have departure times greater than the latest possible departure time, or smaller than the earliest;
- we explicitly model the fact that the salesman has to wait at least $f_B(b)$ time in each station $b \in B$ by introducing a set of *busy-edges*, denoted by $Busy(b)$. We introduce a busy-edge from each arrival node in a station $b \in B$ to the first possible departure node at the same station that differs in the time value by at least $f_B(b)$.
- to model the fact that the salesman starts his journey at some station σ_0 and at time t_0, we introduce a *source node* s_0 in station σ_0 with time value t_0. Node s_0 is connected to the firs departure node d_0 of σ_0 that has a time value greater or equal than t_0, using an edge (called *source edge*) with cost equal to the *cycle-difference*(t_0, t_{d_0}). In addition, we introduce a *sink node* s_f in the same station and we connect each arrival node of σ_0 with a zero-cost edge (called a *sink edge*) to s_f.

4 The Ant Colony Algorithm for Solving the RTSP

In [8], Pintea *et al.* proposed a new algorithm called *Reinforcing Ant Colony System* (*RACS*), based on *Ant Colony System*, in order to solve the *Generalized Traveling Salesman Problem*.

For solving the *RTSP*, the *RACS* algorithm should be changed following the characteristics of the railway problem: the salesman starts from a specific station (cluster), has to visit a given number of stations (cities), spend a minimum amount of time in each station and finally the salesman must return to the starting station.

In what it follows we describe the new algorithm called the *Railway Ant Colony* (*RAC*) algorithm for the *Railway Traveling Salesman Problem*.

In the *RAC* algorithm m ants individually construct candidate solutions in an incremental fashion. The choice of the nest node is based on two main components: pheromone trails and a heuristic value called visibility. The algorithm works as follows:

- Initially the ants are placed at a specified node from the chosen starting cluster (departure station) and all the edges are initialized with a certain amount of pheromone;
- At iteration $t + 1$ every ant moves to a new node from an unvisited *cluster* and the parameters controlling the algorithm are updated;
- Each edge is labelled by a trail intensity. Let $\tau_{ij}(t)$ represent the trail intensity of the edge (i, j) at time t, where i is the starting node and j is the arrival node from an unvisited cluster with the following constraints: the arrival time should be greater than the starting time and also considering the staying time in the cluster. An ant decides in which arrival node is the next move with a probability that is based on the time and the amount of trail intensity on the connecting edge. The inverse of time from a starting node to the next arrival node is known as the *visibility*, η_{ij}.
- At each time unit evaporation takes place. This is to stop the intensity trails increasing unbounded. The evaporation rate is denoted by ρ, and its value is between 0 and 1. In order to avoid ants visiting the same *cluster* in the same tour a tabu list is maintained. This prevents ants visiting *clusters* they have previously visited and therefore ensures the construction of valid tours. The ant tabu list is cleared after each completed tour.
- To favor the selection of an edge that has a high pheromone value τ, and high visibility value η, a probability function ${p^k}_{iu}$ is considered. We denote by ${J^k}_i$ the unvisited neighbors of node i by ant k and if $u \in {J^k}_i$, we denote $u = V_k(y)$, where y is a node from the unvisited cluster V_k. This probability function is defined as follows:

$$ {p^k}_{iu}(t) = \frac{[\tau_{iu}(t)]^{\alpha}[\eta_{iu}(t)]^{\beta}}{\Sigma_{o \in {J^k}_i}[\tau_{io}(t)]^{\alpha}[\eta_{io}(t)]^{\beta}}, \tag{1} $$

where α and β are parameters used for tuning the relative importance of edge time in selecting the next node. ${p^k}_{iu}$ is the probability of choosing $j = u$,

where $u = V_k(y)$ is the next node, if $q > q_0$ (the current node is i). If $q \leq q_0$ the next node j is chosen as follows:

$$j = argmax_{u \in J_i^k}\{[\tau_{iu}(t)]^\alpha [\eta_{iu}(t)]^\beta\}, \tag{2}$$

where q is a random variable uniformly distributed over $[0, 1]$ and q_0 is a parameter similar to the temperature in simulated annealing, $0 \leq q_0 \leq 1$.

– The trail intensity is locally updated using the correction rule:

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \rho \cdot \tau_0. \tag{3}$$

– After each transition the trail intensity is globally updated using the correction rule:

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \rho \frac{1}{n \cdot L^+}. \tag{4}$$

where L^+ is the time of the best tour and n is the total number of nodes.

– In *Ant Colony System* only the ant that generate the best tour is allowed to *globally* update the pheromone. The global update rule is applied to the edges belonging to the *best tour*. The correction rule is

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \rho \Delta\tau_{ij}(t), \tag{5}$$

where $\Delta\tau_{ij}(t)$ is the inverse time of the best tour.

At the beginning, the ants are in their nest and will start to search food in a specific area. An ant goes exactly once through a given number of clusters, B and returns to the nest. An ant stays in each cluster for a period of time, in order to find a desirable food and after that, starting from another node, it is continuing the tour. The choice of the next node is probabilistic. The more pheromone there is on a certain road (edge) the bigger the chance that road will be taken. Therefore, the pheromone trails will guide the ants to the shorter path, a solution of *RTSP*.

When each ant has completed a tour, the amounts of pheromone are updated. On every road (edge) some fraction of pheromone evaporates while on roads, that have been visited by at least one ant new pheromones are deposited. All roads on the best tour so far get an extra amount of pheromone. This is called an elitist ant approach: the best ants reinforce their tour even more.

The process goes on until a certain condition, such as a certain number of iterations, amount of CPU time or solution quality, has been achieved.

The *Railways Ant Colony* algorithm can be stated as follows:

Railways Ant Colony algorithm for the *RTSP*

```
For every edge (i, j) do τ_ij(0) = τ_0
for k = 1 to m do
   place ant k on a specified chosen node from a specified cluster
end for
```

let T^+ be the shortest tour found and L^+ its length
for $iter = 1$ to Nb_{iter} do
  for $k = 1$ to m do
    build tour $T^k(t)$ by applying $B - 1$ *times*
      choose the next *arrival* node j from an unvisited cluster,

$$j = \begin{cases} argmax_{u \in J_i^k}[\tau_{iu}(t)]^\alpha [\eta_{iu}(t)]^\beta & \text{if } q \leq q_0 \\ J & \text{if } q > q_0 \end{cases}$$

      where $J \in J_i^k$ is chosen with probability:

$$p^k{}_{ij}(t) = \frac{[\tau_{iu}(t)]^\alpha [\eta_{iu}(t)]^\beta}{\Sigma_{o \in J^k{}_i}[\tau_{io}(t)]^\alpha [\eta_{io}(t)]^\beta}$$

      and where i is the current *starting* node
      update pheromone trails by applying the local rule:

$$\tau_{ij}(t+1) = (1-\rho) \cdot \tau_{ij}(t) + \rho \cdot \tau_0$$

    end for
  for $k = 1$ to m do
    compute the length $L^k(t)$ of the tour $T^k(t)$
  if an improved tour is found then update $T^k(t)$ and $L^k(t)$
  for every edge $(i,j) \in T^+$ do
    update pheromone trails by applying the global rule:

$$\tau_{ij}(t+1) = (1-\rho) \cdot \tau_{ij}(t) + \rho \cdot \Delta\tau_{ij}(t)$$

     where $\Delta\tau_{ij}(t) = \frac{1}{L^+}$
   end for
end for
Print the shortest tour T^+ and its length L^+

where by Nb_{iter} we denoted the number of iterations.

To evaluate the performance of the proposed algorithm, the *RAC* is going to be compared with the results obtained by Hadjicharalambous *et al.* in [7].

5 Implementation Details and Computational Results

Our proposed *RAC* algorithm have been implemented in Java. The initial value of all pheromone trails, $\tau_0 = 0.1$. The parameters for the algorithm are critical as in all other ant systems. Currently there is no mathematical analysis developed to give the optimal parameter setting in each situation. In the *RAC* algorithm the values of the parameters were chosen as follows: $\alpha = 1$, $\beta = 2$, $\rho = 0.001$, $q_0 = 0.5$ as in [2,3]. There are used 10 ants with 100 iterations for each ant and the maximal number of cycles is 2500.

The experiments were conducted on synthetic as well on real-world data. For the synthetic case we consider the grid graphs constructed in [7]. Each node of the graph has connections with all of its neighboring nodes, i.e., the stations that are located immediately next to it in its row or column in the grid.

The connections among the stations were placed at random among neighboring stations, such that there is at least one connection in every pair of neighboring stations and the average number of elementary connections is 10. The time-differences between the departure and arrival time of each elementary connection are independent uniform random variables, chosen in the interval $[20, 60]$ (representing minutes), while the departure times are random variables in the time interval between the earliest and the latest possible departure time. The number of stations varies from 20 to 200.

The real-world data represent part of the railroad network of the Netherlands. The first data set, called *nd_ic*, contains the Intercity train connections among the larger cities in the Netherlands stopping only at the main train stations. These trains operate at least every half an our.

In Table 1, we present the parameters of the time expanded graph in each data set.

Table 1. Graph parameters for the time expanded graph in each data set

Data Set	Number of Stations	Number of Nodes	Number of Edges	Number of Connections	Conn/ Stations
nd_ic	21	684	1368	342	16.29
nd_loc	23	778	1556	389	16.91
synthetic1	20	400	800	200	10
synthetic2	30	600	1200	300	10
synthetic3	40	800	1600	400	10
synthetic4	50	1000	2000	500	10
synthetic5	60	1200	2400	600	10
synthetic6	70	1400	2800	700	10
synthetic7	80	1600	3200	800	10
synthetic8	100	2000	4000	1000	10
synthetic9	125	2500	5000	1250	10
synthetic10	150	3000	6000	1500	10
synthetic11	175	3500	7000	1750	10
synthetic12	200	4000	8000	2000	10

The first nine rows contain the same data sets as in [7] and the last five columns contain synthetic data constructed by us.

For each data set, several problem instances were created, varying the number $|B|$ of the selected stations, i.e., stations that the salesman has to visit. For both graphs based on real and synthetic data, we used two values for $|B|$, namely 5 and 10. Note that B does not contain the starting station. The stations that belong to B were selected randomly and independent of each other.

For each combination of data set and a value of B, we have selected the stations that belong to B randomly and independently of each other. The selection of the stations have been repeated many times, and the mean values among all corresponding instances were computed.

Each instance we have created was solved by the RAC algorithm. The solutions that we obtained have the same quality with the optimal solutions obtained Hadjicharalambous *et al.* using GLPSOLVE v.4.6 for solving the corresponding integer linear program.

In the next table, we compare the computational results for solving the $RTSP$ using the RAC algorithm with the computational results of Hadjicharalambous *et al.* from [7], in the case of original graphs and reduced graphs.

Table 2. Computational results for $RTSP$ using the RAC algorithm

Data set	$\|B\| = 5$			$\|B\| = 10$		
	CPU RAC	*CPU [7]*	*CPU [7] red.*	*CPU RAC*	*CPU [7]*	*CPU [7] red.*
nd_ic	16.60	-	29.1	374.28	-	6942.6
nd_loc	18.68	-	319.0	677.36	-	9111.9
synthetic 1	5.43	13.12	1.12	78.04	781.12	214.76
synthetic 2	5.51	32.24	1.12	99.95	1287.00	369.59
synthetic 3	8.38	72.06	1.50	119.54	16239.80	214.18
synthetic 4	6.47	-	0.80	132.66	-	181.85
synthetic 5	8.11	-	1.45	152.59	-	257.96
synthetic 6	8.74	-	1.30	189.80	-	431.80
synthetic 7	9.95	-	1.00	196.76	-	233.26
synthetic 8	34.04	-	-	499.90	-	-
synthetic 9	40.56	-	-	613.42	-	-
synthetic 10	48.20	-	-	746.06	-	-
synthetic 11	54.48	-	-	845.28	-	-
synthetic 12	61.02	-	-	986.46	-	-

The columns in the Table 2 are as follows: the first column contains the data sets used in our experiments. The first two rows, called nd_ic and nd_loc, are real data representing parts of of the rail network of the Netherlands. The next rows represent the synthetic data constructed on grid graphs. The next columns contain the number of the selected stations (clusters) $|B|$ that should be in a tour. In the table the sign '-' means that the corresponding information was not provided in [7].

For both graphs based on real and synthetic data, we used two values for $|B|$, namely 5 and 10. For each combination of data set and a value of B, we provided the running times obtained using the RAC algorithm, in comparison with the running times obtained by Hadjicharalambous *et al.* from [7], in the case of original graphs and reduced graphs. The computational values are the result of the average of 50 successively executions of the RAC algorithm.

As we can see from the previous table the *Railway Ant Colony* algorithm performed very well comparing to the results obtained by solving the corresponding integer program from [7]. The average time of 50 runs are used for all values.

The *RAC* algorithm for the *Railway Traveling Salesman Problem* can be improved if more appropriate values for the parameters are used. Also, an efficient combination of *RAC* with other algorithms can potentially improve the results.

6 Conclusion

The paper reveals some interesting aspects of using ant colony in transportation problems. For the *Railway Traveling Salesman Problem* ant algorithms as a robust meta-heuristic has good results compared with integer linear programming solvers [7] for original and reduced graphs. The efficiency of ant-based technique for original graphs it is comparable and in some cases it is even better than the reduced graph algorithm [7] on some real and artificial sets of data. Using appropriate parameters for the introduced *Railway Ant Colony (RAC)* the results could be improved. Another way to improve the results could be hybrid algorithms, combining *Railway Ant Colony* with other optimization techniques.

References

1. Bonabeau, E., Dorigo, M., Tehraulaz, G.: Swarm intelligence from natural to artificial systems. Oxford, UK: Oxford University Press (1999)
2. Dorigo, M., Di Caro, G.: The ant colony optimization meta-heuristic. In D. Corne, M. Dorigo, F. Glover (Eds.), New ideas in optimization 11-32, London: McGraw-Hill (1999)
3. Dorigo, M., Di Caro, G., Gambardella, L.M.: Ant algorithms for discrete optimization.Artificial Life, 5(2), 137-172 (1999)
4. Dorigo, M., Gambardella, L. M.: Ant Colony System: A cooperative learning approach to the Traveling Salesman Problem, IEEE Trans. Evol. Comp., 1, 53-66 (1997)
5. Fischetti, M., Salazar J.J., Toth, P.: The symmetric generalized traveling salesman polytope, Networks, 26, 113-123, (1995)
6. Fischetti, M., Salazar J.J., Toth, P.: A branch-and-cut algorithm for the symmetric generalized traveling salesman problem, Operations Research, 45, 378-394, (1997)
7. Hadjicharalambous, G., Pop, P., Pyrga, E., Tsaggouris, G., Zaroliagis, C.: The Railway Traveling Salesman Problem, in Algorithmic Methods for Railway Optimization, Proc. ATMOS 2004, to appear in Lecture Notes in Computer Science, Springer-Verlag
8. Pintea, C-M., Pop, P.C., Chira, C.: Reinforcing Ant Colony System for the Generalized Traveling Salesman Problem, Proceedings of the First International Conference Bio-Inspired Computing: Theory and Applications, 245-252, (2006)
9. F. Schultz, D. Wagner and K. Weihe, *Dijkstra's Algorithm On-line: An Empirical Case Study from the Public Railroad Transport*, ACM Journal of Experimental Algorithmics, **5:12**, (2000)

Enhancing a MOACO for Solving the Bi-criteria Pathfinding Problem for a Military Unit in a Realistic Battlefield

A.M. Mora[1], J.J. Merelo[1], C. Millan[2], J. Torrecillas[2], J.L.J. Laredo[1], and P.A. Castillo[1]

[1] Departamento de Arquitectura y Tecnología de Computadores. University of Granada (Spain)
{amorag,jmerelo,juanlu,pedro}@geneura.ugr.es
[2] Mando de Adiestramiento y Doctrina. Spanish Army
{cmillanm,jtorrelo}@et.mde.es

Abstract. CHAC, a Multi-Objective Ant Colony Optimization (MOACO), has been designed to solve the problem of finding the path that minimizes resources while maximizing safety for a military unit. The new version presented in this paper takes into acount new, more realistic, conditions and constraints. CHAC's previously proposed transition rules have been tested in more realistic maps. In addition some improvements in the implementation have been made, so better solutions are yielded. These solutions are better than a baseline greedy algorithm, and still good from a military point of view.

1 Introduction

The military logistics unit have as main objective to transport items like fuel, medicines, supplies or ammunition to other allied units. This movement must be previously planned by its commander, usually taking into account two main criteria: speed and safety. The choice of a safe path is made when the enemy forces situation is not known, so the unit must move through hidden zones in order to avoid detection. On the other hand, the choice of a fast path is made when the unit priority is to get to the destination point as soon as possible because of requirements of its mission (for example, strong necessity of items at destination point). But in both situations it is necessary to take into account the other criteria, so in safe path there is a time limit and in fast path the unit must get to destination with enough effectives and supplies to help the other units.

Our algorithm tries to provide this bi-criteria decision by solving what we have called the *military unit pathfinding problem*. This problem is similar to a common pathfinding problem with some additional features and restrictions. It intends to find the best path from an origin to a destination point in the battlefield but keeping a balance between route speed and safety. The unit has an energy (health) and a resource level which are consumed when it moves through the path, so the problem objectives are adapted to minimize the consumption

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 712–721, 2007.
© Springer-Verlag Berlin Heidelberg 2007

of resources and the consumption of energy. In addition, there might be some enemies in the map (battlefield) which could shoot their weapons against the unit. This problem is a particular example of a more general logistics problem, the deployment planning problem [1]. We describe our approach to solve it in [8] using a Multiobjective Ant Colony System called CHAC (in [3,4,5] these concepts are explained). But in this paper we have considered some new constraints and conditions, and a new modelling for the map in order to make the problem more realistic. Moreover, we have changed and added some algorithm features to search for the solutions in this new situation.

2 Problem Features

To solve this problem, the battlefield has been implemented as a grid of hexagonal cells in order to have a more realistic model than in the previous version. Every cell corresponds to a 500x500 meters zone in real world (the same as a real deployed unit).

The problem unit has two properties: a number of *energy points* which represent global health of soldiers or status of vehicles, and it is also the status of the supplies (ammunition, food, fuel or medicins) in this case. The other property is a number of *resource points* which represent unit own supplies such as fuel, food and even moral.

Every cell has assigned a penalization *cost in resources* which represents the difficulty of going through it, and a penalization *cost in energy* which means the unit depletes its human resources or vehicles suffer damage when crossing over the cell (*no combat casualties*). Both costs depend on the cell type. In addition there is another penalization in each case: one in resources if the unit moves between cells with different heights (more if goes up) and other in energy, *lethality*, which is the damage that a cell could produce due to enemy weapons impact.

We speak about fast paths (considering constant velocity) when going through them are little expensive in total cost in resources (is not very difficult to travel through the cells, so it takes little time). Safe paths on the other hand are secure for the unit (little cost in energy).

The types and subtypes of cells, height rank as well as the lethality level, are the same as in previous version [8].

There are two new problem restrictions to consider in the algorithm: the units cannot move through two cells if the height difference between them is greater than 2. It is known as *natural obstacle* (artificial obstacles can be placed in the map as a type of cell). The other restriction is the consideration of the units *adquisition capability* (property of problem unit and enemy ones), which is the longest distance a unit can see. This is a limit for the enemies line of sight and a limit for the exploring radius of the problem unit when there is no known enemy in the map. This distance is about 9 kilometers in real world for every unit, which corresponds to 18 cells of radius in our modelled map. The other problem constraints remains the same.

We have improved the preceding application (in Delphi 7 for Windows XP) including these features. We can also manage a real world battlefield, but it is necessary to define an underlying *information layer* by assigning type, subtype, height and lethality (if wanted) for each cell. We can place the enemies, obstacles, origin and destination points to define completely the problem to solve.

Figure 1 shows an example of real world battlefield and the information layer associated to it.

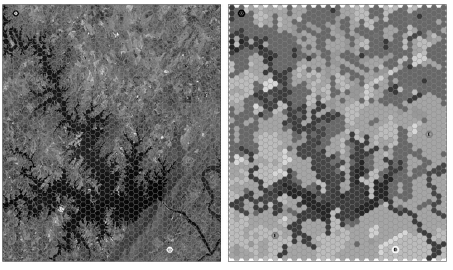


Fig. 1. Example Map (45x45 cells). Right image is a real world picture of a lake surrounded by some hills and lots of vegetation. Left image shows its associated information layer, where it can be seen the types corresponding to the same hexagons in the other image, so there are many water and forest cells and some normal terrain cells. The different shades in the same color models height (light color) and depth (dark color). There are two enemies labelled with 'E', an origin point (in the top-left corner of the images) labelled with 'O' and a destination point (in the bottom-right) labelled with 'D'. These labels are black on the image at right and white on the left.

3 Hexa-CHAC Algorithm

Hexa-CHAC is an Ant Colony System (ACS) adapted to deal with several objectives, that is, a Multi-objective Ant Colony Optimization (MOACO) algorithm (can see a review of some of them in [6]). The Hexa prefix refers to the new topology of the search space (a grid of hexagons).

The problem must be transformed into a graph where each node corresponds to a cell in the map and an edge between two nodes is the existent connection between neighbour cells in the map. Every edge has associated two weights which are the costs in resources and energy that going through that edge causes to the unit. This time, every node has 6 neighbours because of the hexagonal grid topology.

Due to the algorithm is an ACO one [4,5], in each iteration, every ant builds a complete path (solution), if possible, between origin and destination points by travelling through the graph. To guide this movement it uses a state transition rule which combines two kinds of information: pheromone trails and heuristic knowledge.

The problem we want to solve is a multiobjective one [3] with two independent objectives to minimize. These objectives are named $\boldsymbol{f}$, minimization the resources consumed in the path (speed maximization) and $\boldsymbol{s}$, minimization the energy consumed in the path (safety maximization).

So hexa-CHAC (hCHAC on forward) uses two pheromone matrices and two heuristics functions each pair dedicated to one objective, and a single colony. We use an Ant Colony System (ACS) to have better control in the balance between exploration and exploitation by using the parameter q_0. We have implemented two state transition rules, first one similar to the proposed in [7] (Combined State Transition Rule, CSTR) and second one based on dominance of neighbours (Dominance State Transition Rule, DSTR). The local and global pheromone updating formulas are based in the MACS-VRPTW algorithm proposed in [2], with some changes due to the use of two pheromone matrices. All these equations can be consulted in the previous paper [8].

The new problem restrictions have influence in some of these formulas, so the consideration of natural obstacles limits the feasible neighbourhood for each node (it is possible to discard a node if the height difference with its neighbour is greater than 2. It is also affected by new grid topology). The adquisition capability of problem and enemy units change the heuristic functions because it limits the radius in cells that our unit can check and that and enemy can see following the line of sight.

Moreover, we are considering a logistics unit in this paper, so some weights change their values in some formulas to give importance to energy preservation. This way heuristic function for security objective takes greater values in the weights related to visibility and cost in energy. In addition, the weights in the evaluation functions F_f and F_s related to the importance of visibility of cells are also greater. Using all these new values safe paths will be even safer. There is an extra (and new) pheromone updating in the edges of non-completed paths. It is an evaporation proportional to ρ (the common evaporation factor) and it penalizes those paths to avoid that other ants follow them.

At the end of the algorithm we have implemented a new improvement that only applies to the solutions in the Pareto Set. It deletes superfluous cells in paths like long way rounds which means an extra cost in resources and energy. Each improved solution is tested to delete for Pareto Set those solutions which it dominates.

4 Experiments and Results

We have performed experiments in 3 realistic (real terrain) maps, which are part of 3 Panzer General game maps. They are maps of 45x45 cells.

We have used common values for parameters (α=1, β=2, ρ=0.1) excepting q_0=0.4 which value determine more exploration as usual. On the other hand, weights in heuristic and evaluation functions have been changed to tends to secure paths due to the kind of the problem unit. The user, will decide the value for λ parameter, which gives relative importance to one objective over the other,

so if it is near 1, finding fastest path would be more important and if it is near 0, the other way round.

hCHAC yields a set of non-dominated solutions (it is a MO algorithm), but less as usual because it only searches in the region of the ideal Pareto front determined by the λ parameter. In addition, we only consider one (chosen by military participation considering their own criteria and the features of each problem).

We have made 30 executions in each scenario, using each state transition rule and using each extreme value for λ (0.9 and 0.1) in order to find the fastest and the safest path respectively. We are going to compare the results with a greedy approach which uses a *dominance criterion* to select the next node of the actual cell neighbourhood (it selects as next the node which dominates most of the others neighbours considering their cost in both objectives). This algorithm is quite simple, and sometimes does not even reach a solution because it gets into a loop situation.

Map 1 (Single Enemy)

Figure 2 shows this map along with the best solutions found. Cells marked with black and white border compose the solution; those with black border are seen by enemy and those with white border are hidden. We have labelled in white: origin and destination cells with 'O' and 'D' respectively, and enemy unit with 'E'.

Table 1. Results for Map 1 (1500 iterations, 50 ants)

	Combined State Transition Rule				*Dominance State Transition Rule*				*Greedy*
	Fastest (λ=0.9)		Safest (λ=0.1)		Fastest (λ=0.9)		Safest (λ=0.1)		
	F_f	F_s	F_f	F_s	F_f	F_s	F_f	F_s	
Best	**68.50**	295.40	80.50	**7.30**	**76.00**	306.10	95.50	**9.40**	
Mean	**75.20**	184.54	85.00	**8.10**	**81.63**	271.11	108.00	**10.40**	—
	±**7.87**	±132.49	±3.32	±**0.49**	±**3.02**	±39.98	±5.70	±**0.52**	

As it can be seen in Table 1, CSRT yields better results than DSTR in terms of best solution and mean value, but these values are close each other. It occurs due to DSTR has a higher exploration component because of the formula it uses, but in an hexagonal grid there are less possibilities of move in each node (less neighbours) than in the previous grid (where there were 8 neighbours), so results have improved for this rule. Both methods yield results with a low standard deviation, which means the algorithms have robustness (its solutions are similar between executions), DSTR is even better sometimes in both objectives, so it may be more robust. Best, mean and standard deviation in the objective which is not being minimized are logically worse, because it has little importance. But in this case the differences between the security cost (F_s) when it is prioritary and secondary objective are enormous. The reason is fast paths are usually unsafe due to visibility of the cells, and in this experiments we penalize much the term of visibility in the cost function because energy is very important for logistics

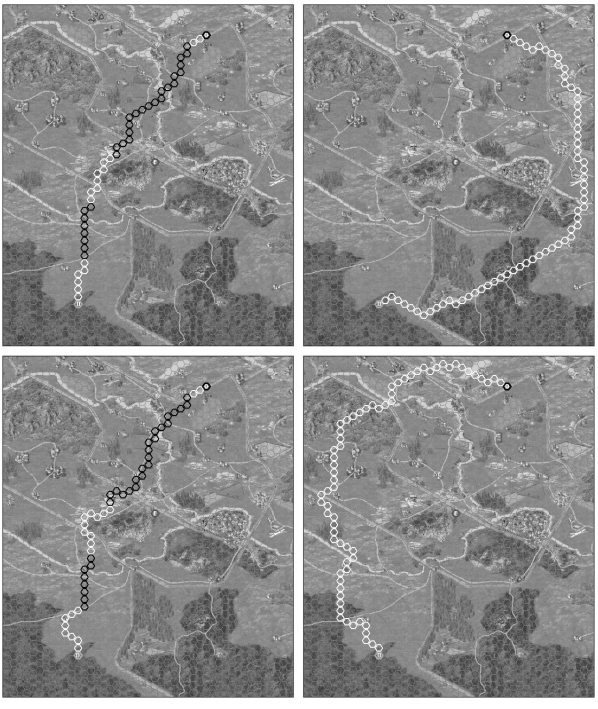


Fig. 2. Map 1. CSTR results (top) and DSTR (bottom); fastest (left) and safest (right).

units as we previously said. There is no solution for Greedy algorithm, so it cannot be compared.

In Figure 2 (top-left) we can see the fastest path found with CSTR, the unit goes in a rather straight way to the destination point, in bottom-left image the fastest path for DSTR is showed and we can see that it is similar, but it is less straight, it has more cells so the cost in boths objectives are worse. Both paths have seen cells because they go near the enemy but it also have hidden ones because it moves through forest or behind hills, the reason is security is important even in searching for fast paths. Figure 2 top-right and bottom-right show the safest path found with CSTR and DSTR respectively. Both of them represents a curve (distance to target point has little importance) which increases speed cost, but the unit goes through all hidden cells. Each solution moves by a zone of the map, but they two surround the enemy adquisition capability area and go through some forest cells or through mountains to avoid to be seen (safe cells). This behaviour is excellent from military tactical point of view.

Map 2 (No Enemy)

Figure 3 shows this map along with the best solutions found by each method of hCHAC. Cells marked with black border compose the solution, because there are no absolutely seen or hidden cells (there is no enemy watching over); the labels are the same as before.



Fig. 3. Map 2. CSTR results (top) and DSTR (bottom); fastest (left) and safest (right).

Table 2 shows results obtained by hCHAC in this map; where again standard deviation is low. The most important fact is the high value for security cost (F_s) in all cases. It occurs due to no enemy is located, so there are no hidden cells (all are seen in some measure) which means the security cost is dramatically increased. Again CSTR and DSTR yielded similar values. In addition, all the solutions have similar costs in both objectives because a straight path to destination point is a good result in both (it moves through mountain cells which are often hidden) as we can see in Figure 3. For the same reason, the greedy algorithm yields a solution very close to CSTR.

If we look at cost in resources (F_f) CSTR is better too. On the other hand, greedy is better than the best DSTR solution, because the experiment using this rule may need a little more exploitation in search.

In Figure 3 we can see the best solutions found by hCHAC using CSTR (top) and DSTR (bottom), which are very similar, but safest paths (right) moves near or even through hills and mountains to hide from other cells. The fastest CSTR path (top-left) goes around the mountain at top because it is a natural obstacle in its south-east side, the other paths 'climb' it for its south-west side where it is feasible. Again, all of this actions correspond to a sensible military behaviour.

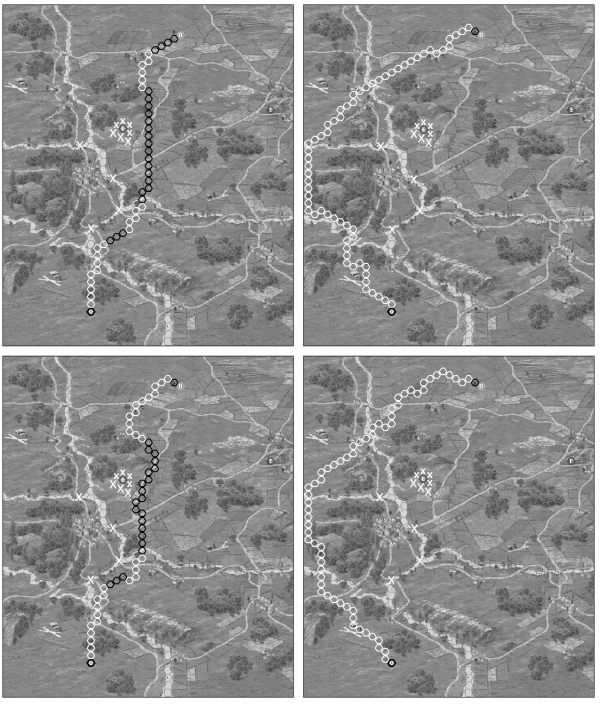


Fig. 4. Map 3. CSTR results (top) and DSTR (bottom); fastest (left) and safest (right).

Table 2. Results for Map 2 (1500 iterations, 50 ants)

	Combined State Transition Rule				*Dominance State Transition Rule*				*Greedy*	
	Fastest (λ=0.9)		Safest (λ=0.1)		Fastest (λ=0.9)		Safest (λ=0.1)			
	F_f	F_s	F_f	F_s	F_f	F_s	F_f	F_s	F_f	F_s
Best	**75.15**	348.58	80.59	**335.28**	**79.36**	352.77	82.74	**350.21**		
Mean	**77.26**	352.03	80.53	**344.96**	**86.26**	371.02	86.70	**368.81**	81.99	335.5
	±**1.58**	±5.68	±1.98	±**4.51**	±**3.52**	±10.37	±3.50	±**9.71**		

Table 3. Results for Map 3 (1500 iterations, 50 ants)

	Combined State Transition Rule				*Dominance State Transition Rule*				*Greedy*
	Fastest (λ=0.9)		Safest (λ=0.9)		Fastest (λ=0.9)		Safest (λ=0.9)		
	F_f	F_s	F_f	F_s	F_f	F_s	F_f	F_s	
Best	**61.00**	244.90	74.00	**27.30**	**67.50**	235.60	82.50	**28.00**	
Mean	**66.42**	225.19	84.68	**28.36**	**72.92**	236.97	95.93	**29.43**	—
	±**3.29**	±90.26	±4.89	±**0.48**	±**2.63**	±42.74	±7.25	±**0.72**	

Map 3 (Two Enemies Firing)

Figure 4 shows the map along the best solutions found by each method of hCHAC. Cells with black border are seen by any enemy and those with white

border are hiden to both of them. The labels are the same as before, but the 'X' (vertical) marks the enemy weapons impact zone (have an associated lethality).

As it can be seen in Table 3, the results of CSTR and DSTR are again close, and the standard deviation are anew low. The greedy algorithm cannot yield solution.

Fast paths (Figure 4, left) are quite straight as above, but avoid crossing the four bridges the armed enemy is watching, and the surroundings of the enemy unit to avoid increasing the security cost (F_s) with the lethality associated to these cells. Even so this cost is high due to the visibility of central cells which both enemies can see. *Safe* paths (Figure 4, right) move through forest and mountains (hidden to enemy units), and take a detour to avoid the visibility zone of the closest enemy. DSTR solutions are slightly worse, probably due to higher exploration factor.

5 Conclusions and Future Work

This paper describes an enhanced version of our MOACO algorithm (CHAC) called hCHAC, which finds the fastest and safest path (with relative importance set by the user) for a simulated military unit. The improvements include modelling scenarios as a grid of hexagons and the possibility of work with real world images by defining an underlying information layer. In addition new constraints and conditions are considered: natural obstacles that the unit cannot go through, limited unit adquisition capability and focus on a logistics unit, so security is more important (to preserve the supplies which it transports). All these features make the problem more realistic. Finally we have modified the algorithm by fine-tuning solutions and a new penalty factor for incomplete solutions, which accounts for the better solutions found.

hCHAC also uses two different state transition rules: the first one combines heuristic and pheromone information of two objectives (Combined State Transition Rule, CSTR) and the second one is based on dominance over neighbours (Dominance State Transition Rule, DSTR). Both have been tested over several different, and realistic, scenarios, yielding very good results in a subjective assessment by the military staff of the project and being perfectly compatible with military tactics. In complicated maps it offers good results in less time than an expert, and outperforming a greedy algorithm. In the comparison between them, hCHAC with CSTR yields better results in the same conditions, but hCHAC with DSTR is more robust, yielding solutions that perform similarly, so that, by increasing exploitation level or the number of iterations, DSTR might approach the results obtained by DSTR.

Acknowledgements

The authors acknowledge the support of the SIMAUTAVA and NADEWeb (TIC2003-09481-C04) projects.

References

1. Ibrahim Akgün and Barbaros Ç. Tansel. Optimization of transportation requirements in the deployment of military units. *In Computers & Operations Research*, Volume 34, Issue 4, pages 1158–1176, 2005.
2. B. Barán and M. Schaerer. A multiobjective ant colony system for vehicle routing problem with time windows. In *IASTED International Multi-Conference on Applied Informatics*, number 21 in IASTED IMCAI, pages 97–102, 2003. http://www.scopus.com/scopus/inward/record.url?eid=2-s2.0-1442302509&partner4O&relR4.5.0.
3. Carlos A. Coello Coello, David A. Van Veldhuizen, and Gary B. Lamont. *Evolutionary Algorithms for Solving Multi-Objective Problems*. Kluwer Academic Publishers, 2002.
4. M. Dorigo and G. Di Caro. The ant colony optimization meta-heuristic. In D. Corne, M. Dorigo, and F. Glover, editors, *New Ideas in Optimization*, pages 11–32. McGraw-Hill, 1999.
5. M. Dorigo and T. Stützle. The ant colony optimization metaheuristic: Algorithms, applications, and advances. In G.A. Kochenberger F. Glover, editor, *Handbook of Metaheuristics*, pages 251–285. Kluwer, 2002.
6. C. García-Martínez, O. Cordón, and F.Herrera. An empirical analysis of multiple objective ant colony optimization algorithms for the bi-criteria TSP. In *ANTS 2004. Fourth International Workshop on. Ant Colony Optimization and Swarm Intelligence*, number 3172 in LNCS, pages 61–72. Springer, 2004.
7. Steffen Iredi, Daniel Merkle, and Martin Middendorf. Bi-criterion optimization with multi colony ant algorithms. In E. Zitzler, K. Deb, L. Thiele, C. A. Coello Coello, and D. Corne, editors, *Proceedings of the First International Conference on Evolutionary Multi-Criterion Optimization (EMO 2001)*, volume 1993 of *Lecture Notes in Computer Science*, pages 359–372, Berlin, 2001. Springer-Verlag.
8. A.M. Mora, J.J. Merelo, C. Millan, J. Torrecillas and J.L.J. Laredo. CHAC. A MOACO Algorithm for Computation of Bi-Criteria Military Unit Path in the Battlefield. In *Proceedings of the First Workshop in Nature Inspired Cooperative Strategies for Optimization (NICSO'06)*, pages 85–98, Granada, 2006. http://www.citebase.org/abstract?id=oai:arXiv.org:cs/0610113

GRASP with Path Relinking for the Capacitated Arc Routing Problem with Time Windows

Mohamed Reghioui*, Christian Prins, and Nacima Labadi

ICD - OSI, University of Technology of Troyes,
12, Rue Marie Curie, BP 2060, 10010 Troyes France
{nacima.labadi,christian.prins,mohamed.reghioui_hamzaoui}@utt.fr
http://www.utt.fr/labos/LOSI

Abstract. A greedy randomized adaptive search procedure with path relinking is presented for the capacitated arc routing problem with time windows. Contrary to the vehicle routing problem with time windows, this problem received little attention. Numerical experiments indicate that the proposed metaheuristic is competitive with the best published algorithms: on a set of 24 instances, it reaches the optimum 17 times and improves the best-known solution 5 times, including 4 new optima.

Keywords: arc routing, time windows, GRASP, path relinking.

1 Introduction

The NP-hard capacitated arc routing problem (CARP) is raised by applications like urban waste collection and winter gritting. A natural extension is the CARP with time windows or CARPTW, motivated for instance by the interruption of operations during rush hours or maintenance interventions on roads.

The CARPTW can be defined on an undirected network $G = (V, E)$. The set of nodes V includes a depot node s with a fleet of identical vehicles of capacity Q. Each edge $e \in E$ has a positive demand d_e, a length c_e, a deadheading time t_e (traversal time if the edge is not serviced) and a processing time p_e (when it is traversed to be serviced). The m edges with non-zero demands (*required edges*) have also a time window $[a_e, b_e]$, where a_e is the earliest service time and b_e the latest service time. Arriving earlier than a_e induces a waiting time while arriving later than b_e is not allowed. The CARPTW consists of computing a set of vehicle trips minimizing the total distance travelled. Each trip must start and end at the depot and service a subset of required edges while respecting their time windows. The total demand satisfied by a vehicle must not exceed its capacity and each required edge must be serviced by one single vehicle (split service is not allowed).

The total distance is minimized in this paper because this is the usual objective for the vehicle routing with time windows (VRPTW). The algorithm proposed can easily be adapted to minimize the total duration of trips.

* Corresponding author.

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 722–731, 2007.
© Springer-Verlag Berlin Heidelberg 2007

To the best of our knowledge, the only works that deal with the CARPTW are three Ph.D. dissertations. Mullaseril [8] considered the directed CARPTW and proposed heuristics and a transformation into a VRPTW. Later, Gueguen [3] described an integer linear programming model for the undirected CARPTW and another transformation into a VRPTW, but without numerical results. More recently, Wohlk [12] presented two ways of modeling the undirected CARPTW (an arc routing and a node routing formulation), various heuristics and a dynamic programming algorithm combined with simulated annealing (DYPSA). She also designed a column generation method to get good lower bounds.

Tagmouti et al. [11] studied a closely related problem, the CARP with time-dependent service costs, with a column generation method based on a node routing representation. However, such a problem involves time-dependent piecewise-linear service cost functions rather than hard time windows.

Apart from the SA embedded in Wohlk's DP method, no metaheuristic has been proposed for the CARPTW, and only one published journal paper has applied path relinking to a vehicle routing problem [4]. This paper bridges the gap by presenting a simple and effective greedy randomized adaptive search procedure (GRASP) combined with path relinking (PR). It is structured as follows. The main components of GRASP and PR are presented respectively in sections 2 and 3. In section 4, numerical tests to evaluate the effectiveness of the approach are reported. Finally, some conclusions are given in section 5.

2 Components and General Structure of the GRASP

GRASP, introduced in the early nineties by Feo and Resende [1], has been applied to many combinatorial optimization problems. Compared to other metaheuristics like tabu search and genetic algorithms, it offers a good compromise between solution quality and running time.

Each iteration computes one solution in two phases. The first phase builds one trial solution using a greedy randomized heuristic. Each step of this heuristic extends a partial solution by adding one edge randomly selected among the best candidates (restricted candidate list or RCL). The second phase improves the trial solution with a local search procedure. The method stops after a fixed number of iterations and returns the best solution found. The algorithms for the two phases are presented in the two following subsections.

2.1 Greedy Randomized Heuristics

It is not an easy task to design a greedy randomized heuristic able to provide good but diverse solutions. On one hand, the same trial solution can be generated several times if the RCL is too small (the heuristic tends to behave like a deterministic greedy heuristic). On the other hand, a large RCL leads to a random heuristic with low-quality solutions on average. Another difficulty is that the local search gives the same local optimum when applied to several trial solutions that belong to the same attraction basin, and the probability to obtain duplicate solutions after improvement increases with the number of GRASP iterations.

For a given number of GRASP iterations, a better diversity can be expected if several constructive heuristics are used and this was confirmed during preliminary testing. For the CARPTW, two heuristics described in the sequel were selected: a randomized path-scanning algorithm with additional criteria, and a randomized route-first/cluster-second method. In practice, each heuristic is used during one half of GRASP iterations. Some results which prove the effectiveness of this strategy are given in the fifth section.

Randomized Path-Scanning Heuristic. Path-scanning is a greedy heuristic for the CARP proposed by Golden et al. [2]. It builds one trip at a time. At each iteration, the current trip is extended by adding the most promising edge e according to one greedy criterion. The vehicle returns to the depot when no additional edge fits its residual capacity. Five criteria are used to select e: 1) minimize the distance to return to the depot, 2) maximize this distance, 3) minimize the ratio d_e/c_e (a kind of productivity), 4) maximize this productivity, 5) use criterion 2 if vehicle is less than half-full, otherwise use criterion 1. One solution is computed for each criterion and the best one is returned at the end. For the same computing time (construction of 5 solutions), Pearn [9] noticed that better results are obtained on average if one of the 5 criteria is randomly selected before each addition of an edge to a trip.

For the CARPTW, we added twelve other criteria proposed by Wohlk [12] and better suited for time windows (they include for instance an earliest due date criterion). Like in Pearn, one of the criteria (now 17) is randomly selected at each iteration. A second randomization consists of drawing each added edge from an RCL containing the k best candidates with compatible demands and time windows. Again, some preliminary tests have shown that using all criteria gives better results than using only the classical CARP criteria or the ones specifically designed for time windows.

Randomized Route-First/Cluster-Second Heuristic. This heuristic starts by building one giant tour servicing all required edges, ignoring vehicle capacity and time window constraints. A randomized nearest neighbor method is used for this purpose. Then, the optimal partition (subject to the sequence) of the giant tour into feasible trips is computed using a tour splitting method called *Split*. A similar splitting technique was used successfully by Lacomme et al. to evaluate chromosomes in a memetic algorithm for the CARP without time windows [6].

The upper part of Figure 1 shows a giant tour $T = (a, b, c, d, e)$ servicing five required edges (thick segments), with demands in brackets and time windows in square brackets, assuming $Q = 9$. Thin segments denote shortest paths between edges. *Split* builds an auxiliary graph in which each arc models a feasible trip. For instance, the leftmost arc corresponds to the trip reduced to edge a: its cost 35 includes the distance from the depot to the edge, the length of the edge, and the distance to come back to the depot. Once this auxiliary graph is completed, *Split* computes a min-cost path between the first and last nodes. The arcs along this path indicate the best decomposition of the giant tour (lower part of the figure).

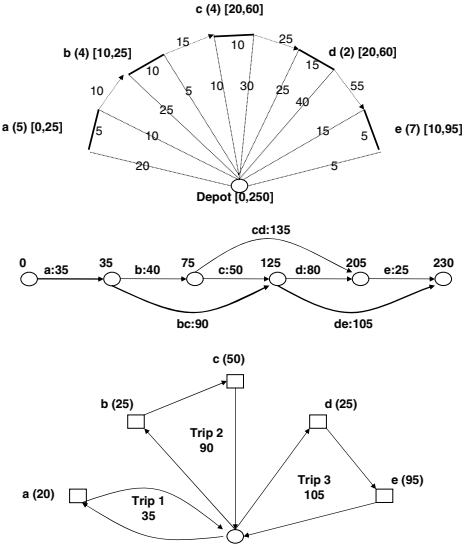


Fig. 1. Split for CARPTW

2.2 Local Search

Possible moves in a local search for arc routing problems include relocating, swapping and inverting chains of edges. Feasibility tests for capacity can be done easily in $O(1)$ but the whole trip must be scanned in $O(m)$ to check time windows. This can be avoided by storing some useful information, as described by Kindervater et al. [5] for the VRPTW. The following moves have been selected:

- OR-OPT: relocate a chain of 1, 2 or 3 required edges.
- SWAP: swap two different edges.
- 2-OPT : replace two shortest paths by two others.

These moves may involve one or two trips, and the two traversal directions of a required edge are tested in the reinsertions. Each edge is represented by two opposite arcs. For one arc $u = (i, j)$, let u' denote the opposite arc (j, i). For instance, an OR-OPT move may remove one arc u from its current trip to reinsert it as u or u', in the same trip or in another trip.

Consider now 2-OPT moves and two pairs of consecutive arcs (u, v) and (w, x). If they belong to the same trip, the 2-OPT move replaces the shortest paths linking u to v and w to x by the shortest paths from u to w' and from v' to x: this corresponds to a classical 2-OPT move for the TSP. If the two pairs belong to distinct trips, there are two different ways of reconnecting the trips, by adding either the paths from u to x and from w to v or the paths from u to w' and from v' to x.

Each iteration of the local search evaluates all feasible moves and executes the best one. The local search ends when no more improvement can be found.

2.3 General Structure of the GRASP

Algorithm 1 shows the pseudo-code of our GRASP, without the PR procedure. At each iteration, the solution S resulting from one of the two heuristics is improved by local search. The algorithm stops after a fixed number of iterations *maxiter* or when a known lower bound LB is reached. Its results are the best solution found S^* and its cost f^*.

Algorithm 1. GRASP for the CARPTW

```
1: f* := ∞
2: iter := 0
3: repeat
4:    iter := iter + 1
5:    if (iter ≤ maxiter/2) then
6:       RandomizedPathScanning(S)
7:    else
8:       RandomizedRouteFirstClusterSecond(S)
9:    end if
10:   LocalSearch(S)
11:   if f(S) < f* then
12:      f* := f(S)
13:      S* := S
14:   end if
15: until (iter = maxiter) or (f* = LB)
```

3 Path Relinking

Path relinking (PR) is an intensification strategy which can be added to any metaheuristic. Its principle is to explore a trajectory that links two solutions in solution space, to try finding a better one. The attributes of the target solution are progressively introduced in the source solution, to generate a sequence of intermediate solutions. At each iteration, the attributes added are selected to minimize the cost of the next intermediate solution.

In spite of its simplicity and speed, GRASP is often less effective than more agressive metaheuristics like tabu search. A possible explanation is the independency of GRASP iterations, which perform a kind of random sampling of solution space. The addition of a PR working on a small pool of elite solutions collected during the GRASP is a good way to remedy this relative weakness. The following subsections describe a distance measure for the CARPTW, show how to use it to explore a trajectory between two solutions, and present the general structure of our algorithm which integrates the PR mechanism into the GRASP.

3.1 Distance Measure

Our distance for the CARPTW extends a distance proposed by Marti et al. [7] for permutation problems, in which the relative position of each element

is more important than its absolute position (R-permutation problems). Given two permutations $S = (p_1, p_2, \ldots, p_m)$ and $T = (q_1, q_2, \ldots, q_m)$, this distance $D(S,T)$ is the number of times p_{i+1} does not immediately follow p_i in T, for $i = 1, 2, \ldots, m-1$. In other words, D is the number of pairs of consecutive elements in P that are "broken" in Q, it varies between 1 and $m-1$.

A similar strategy is used to compute a distance between CARPTW solutions, represented as giant tours to have a constant length of m required edges. The *Split* procedure of 2.1 is used to extract the true CARPTW solutions. We have also extended the original distance to make it reversal-independent: a pair of consecutive arcs (u, v) of S is counted as broken if neither (u, v) nor (v', u') are found in T. Figure 2 provides an example with $D(S,T) = 4$. Vertical bars in S (resp. T) indicate the pairs broken in the other solution.

```
S:  2   5  10 | 7   6 | 8   1 | 9 | 4   3
T:  9 | 6'  7'| 2   5  10 | 8   1 | 4   3
```

Fig. 2. Example of distance for the CARPTW

3.2 Exploration of a Trajectory

To generate a path from one solution S to one solution T, we first compute $D(S,T)$. Then, starting from S, the incumbent solution undergoes a move that reduces its distance to T. The moves include the inversion, the relocation and the relocation with inversion of a chain of consecutive edges without broken pairs, called *block*. The concept of block is used to avoid repairing one pair while breaking another. In Figure 2 for instance, moving arc 1 after arc 9 in S repairs the pair $(1, 4)$ but breaks $(8, 1)$. This can avoided if the move concerns one complete block, here $(8, 1)$.

To avoid too long trajectories and the generation of low-quality solutions, all possible moves are evaluated and the least-cost one is executed to obtain the next solution on the path. This new solution is immediately decoded with *Split*.

It is well-known that the PR mechanism alone provides few improved solutions: it must be helped by a local search. However, consecutive solutions on the path often lead to the same local optimum because their structures are too close. For the CARPTW, the local search already described for the GRASP is applied only every λ iterations, λ being a given parameter. This accelerates the path relinking with a negligible loss in solution quality.

3.3 Integrating GRASP and PR

There are two main ways of combining GRASP and PR. The first one is to use PR in post-optimization, on a small pool of elite solutions collected during the GRASP. The second is to apply it to each local optimum, by exploring the path which links it to an elite solution randomly chosen in the pool. The second strategy was selected because it is more effective according to Resende and Ribeiro [10], and

Algorithm 2. GRASP with PR for the CARPTW

```
1: iter := 0
2: repeat
3:    iter := iter + 1
4:    if (iter ≤ maxiter/2) then
5:       RandomizedPathScanning(S)
6:    else
7:       RandomizedRouteFirstClusterSecond(S)
8:    end if
9:    LocalSearch(S)
10:   if iter = 1 then
11:      P:= {S}; S* := S; f_max := f(S)
12:   else
13:      T = arg max {D(S, X) : X ∈ P}
14:      k := 0
15:      Q := S
16:      while Q ≠ T do
17:         MoveOnPath(Q)
18:         k := k + 1
19:         if (k mod λ) = 0 then
20:            LocalSearch(Q)
21:         end if
22:         if |P| < ρ then
23:            P := P ∪ {Q}
24:         else if f(Q) < f_max then
25:            Y = arg min {D(Q, X) : X ∈ P ∧ f(X) > f(Q)}
26:            P := (P \{Y}) ∪ {Q}
27:            update best solution S* and worst cost f_max
28:         end if
29:      end while
30:   end if
31: until (iter = maxiter) or (f(S*) = LB)
```

this was confirmed by our preliminary testing. Our implementation is sketched in Algorithm 2.

Our PR works on a pool $\mathcal{P}$ limited to ρ solutions. At the first GRASP iteration, the pool is initialized with the first solution. In each subsequent iteration, the PR generates a path between the incumbent GRASP solution S and the solution T most different from S in $\mathcal{P}$.

The incumbent solution Q on the trajectory is modified by *MoveOnPath*. This procedure evaluates all ways of moving or inverting a block in Q and performs the least-cost move among the ones which reduce the distance between Q and T. Remember that distance computations consider solutions encoded as strings without delimiters. The string corresponding to one detailed solution is obtained by concatenating the sequences of arcs traversed by the trips. Conversely, a string can be decoded using the *Split* procedure. These conversions are not detailed in Algorithm 2, to avoid complicating its general structure.

The counter k counts the solutions generated along the path. As explained in previous subsection, the local search is applied only every λ solutions. If $\mathcal{P}$ has not yet reached its nominal size ρ, the intermediate solution Q is added to the pool. Otherwise, Q enters the pool only if it is better than the worst solution in $\mathcal{P}$. f_{max} denotes the cost of this worst solution. To increase pool diversity, the solution Y most similar to Q (in terms of distance) is replaced. This solution is selected among the solutions of $\mathcal{P}$ outperformed by Q. Note that the best pool solution S^* is preserved or improved, but never lost.

4 Computational Results

Three sets of problems A, B, C proposed by Wohlk and containing 8 instances each were used for our tests. These sets mainly differ in the width of their time windows: tight for A, wide for C, and intermediate for B. Our algorithms were implemented in Delphi and executed on a 3 GHz PC, with the following parameters: number of GRASP iterations $maxiter = 500$, pool size $\rho = 8$, local search applied every $\lambda = 2$ solutions on PR trajectories.

The improvement brought by the use of two different greedy randomized heuristics is summarized by Table 1. The three last columns respectively correspond to a GRASP with 500 calls to the Path Scanning heuristic (PS), 500 calls to the Route-First/Cluster Second heuristic (RFCS) and 250 calls to each heuristic (PS+RFCS). The rows provide the average deviation to a lower bound LB in %, the worst deviation and the number of proven optima (when LB is reached). The lower bounds were computed by Wohlk using column generation [12]. The last strategy gives the best results and finds one more optimum.

Table 2 provides a comparison instance per instance between our results and those obtained by Wohlk. Column 2 shows the number of edges for each instance, the 3rd column indicates Wohlk's lower bound. Columns 4, 5 and 6 respectively represent the total distance obtained by the Preferable Neighbor Heuristic (PNH) proposed by Wohlk, the GRASP with one run and the same GRASP with the PR. The last two columns give the average distance obtained by performing 10 runs, and the best cost achieved so far. The last rows give the average deviation to the lower bound in %, the average running time in seconds, and the number of proven optima. Unfortunately, no running times are given for PNH in Wohlk's thesis.

PNH is a kind of two-phase heuristic column generation method. The first phase generates a set of promising feasible tours. Using a commercial IP solver (CPLEX), the second phase solves a set covering problem defined by these routes.

Table 1. Impact of greedy randomized heuristics on the GRASP

	PS	RFCS	PS+RFCS
Avg. dev. LB	3.93%	4.52%	3.76%
Worst dev. LB	18.49%	19.09%	18.49%
Optima	6	6	7

Table 2. Results for each instance

Instance	$\|E\|$	LB	PNH	GRASP	GRASP with PR	GRASP with PR 10 runs	Best GRASP with PR
A10A	15	107	*107	*107	*107	*107	*107
A13A	23	202	*202	*202	*202	*202	*202
A13B	23	171	173	173	***171**	***171**	***171**
A13C	23	163	*163	167	*163	*163	*163
A20B	31	260	264	264	***260**	***260**	***260**
A40C	69	660	*660	706	*660	*660	*660
A40D	69	807	*807	843	812	811.4	*807
A60A	90	1822	*1822	1884	1830	1832.8	1830
B10A	15	87	*87	*87	*87	*87	*87
B13A	23	167	*167	*167	*167	*167	*167
B13B	23	152	158	***152**	***152**	***152**	***152**
B13C	23	141	*141	142	*141	*141	*141
B20B	31	214	*214	*214	*214	*214	*214
B40C	69	588	602	644	602	602	602
B40D	69	730	*730	768	*730	730.3	*730
B60A	90	1554	*1554	1654	1575	1574.8	1565
C10A	15	73	*73	*73	*73	*73	*73
C13A	23	142	148	149	***142**	***142**	***142**
C13B	23	132	*132	135	*132	*132	*132
C13C	23	121	*121	123	*121	*121	*121
C20B	31	186	*186	192	*186	*186	*186
C40C	69	503	563	588	**547**	552.6	**547**
C40D	69	611	626	672	636	636.2	632
C60A	90	1283	*1283	1405	1305	1313.7	1300
Avg. dev. LB			1.20%	3.76%	0.80%	0.87%	0.70%
Optima			17	7	17	16	18
Avg. time			?	6.1s	54.7s	63.4s	

The basic GRASP is very fast but finds few optima. The PR brings a strong improvement at the expense of a running time multiplied by 9. The GRASP with PR finds the same number of optimal solutions as PNH (17 out of 24 instances) but displays a smaller deviation to the lower bound: 0.8% versus 1.2%. Four new optima are found (instances A13B, A20B, B13B, C13A) and the best-known solution for C40C is improved. The column with 10 runs shows that our method is quite stable. By modifying the pool size and the periodicity of local search in the PR, another optimum was achieved (A40D), thus reducing the average deviation to the lower bound to 0.75%.

5 Conclusions

A GRASP for the very hard CARPTW has been reinforced by using two different randomized heuristics and a PR based on a distance measure in solution space.

The resulting algorithm competes with the best existing heuristic: 17 out of 24 instances are solved to optimality and the gap to lower bounds is less than 1%. Moreover, it is relatively simple and requires only 3 parameters.

Like in the VRPTW, the time windows considered concern the service: it is still possible to traverse an edge without servicing it, inside its window. Sometimes, works in a street (e.g. repairing an optic fiber) prevent deadheading traversals too. Our goal is now to tackle such cases, in which the shortest path between two edges is affected by the time windows of traversed edges.

References

1. Feo, T.A., Resende, M.G.C.: Greedy randomized adaptive search procedures. Vol. 6. Journal of Global Optimization (1995) 109–133
2. Golden, B.L., DeArmon, J.S., Baker, E.K.: Computational experiments with algorithms for a class of routing problems. Vol. 10(1). Computers & Operation Research (1983) 47–59
3. Gueguen, C.: Exact solution methods for vehicle routing problems. Ph.D. thesis (in French), Central School of Paris (1999)
4. Ho S.C., Gendreau M.: Path relinking for the vehicle routing problem. Vol. 12(1-2). Journal of Heuristics (2006) 55-72
5. Kindervater, G.A.P., Savelsbergh, M.W.P.: Vehicle routing: handling edge exchanges. In E.H.L. Aarts and J.K. Lenstra (Eds.): Local Search in Combinatorial Optimization, Wiley, Chichester (1997) 337–360
6. Lacomme P., Prins C., Ramdane-Chérif W.: Competitive memetic algorithms for arc routing problems, Vol. 131. Annals of Operations Research (2004) 159–185
7. Marti, R., Laguna, M., Campos, V.: Scatter Search vs Genetic Algorithms : an experimental evaluation with permutation problems. In C. Rego and B. Alidaee (Eds.): Metaheuristic optimization via memory and evolution, Tabu Search and Scatter Search, OR/CS Interfaces Series vol. 30, Springer (2005) 263–283
8. Mullaseril, P.A.: Capacitated Rural Postman Problem with Time Windows and Split Delivery. Ph.D. thesis, MIS Department, University of Arizona, Tucson, Arizona (1997)
9. Pearn, W.L.: Augment-insert algorithms for the Capacitated Arc Routing Problem. Vol. 18(2). Computers & Operations Research (1991) 189–198
10. Resende, M.G.C., Ribeiro, C.C.: GRASP with path-relinking: recent advances and applications. In T. Ibaraki, K. Nonobe and M. Yagiura (Eds.): Metaheuristics: Progress as Real Problem Solvers, Springer (2005) 29–63
11. Tagmouti, M., Gendreau, M., Potvin, J.-Y.: Arc routing problems with time-dependent service costs, Research report 2005/10, CRT, Montréal (2005)
12. Wohlk, S.: Contributions to arc routing. Ph.D. thesis, Faculty of Social Sciences, University of Southern Denmark (2005)

Multi-objective Supply Chain Optimization: An Industrial Case Study

Lionel Amodeo, Haoxun Chen, and Aboubacar El Hadji

ICD - LOSI (FRE CNRS 2732) - University of Technology of Troyes -
12 rue Marie Curie , 10012 Troyes - France
lionel.amodeo@utt.fr, haoxun.chen@utt.fr

Abstract. Supply chain optimization usually involves multiple objectives. In this paper, supply chains are optimized with a multi-objective optimization approach based on genetic algorithm and simulation model. The supply chains are first modeled as batch deterministic and stochastic Petri nets, and a simulation-based optimization method is developed for inventory policies of the supply chains with a multi-objective optimization approach as its search engine. In this method, the performance of a supply chain is evaluated by simulating its Petri net model, and a Non dominated Sorting Genetic Algorithm (NSGA2) is used to guide the optimization search process towards global optima. An application to a real-life supply chain demonstrates that our approach can obtain inventory policies better than ones currently used in practice in terms of two objectives: inventory cost and service level.

Keywords: Supply chain management, Petri Net, Simulation, Multi-objective Optimization, NSGA-II.

1 Introduction

The goal of this study is to develop a practical tool that can help the companies to reduce their inventory costs and improve their customer services by optimizing the inventory policies of their supply chains. Petri nets have been proved a very powerful tool for modeling and analysis of discrete event systems such as manufacturing systems. The tool is also applicable to supply chains since they are discrete event systems as well from a high level of abstraction. In the literature supply chains are usually described as multi-echelon inventory systems [8] but most existing models can only describe a restricted class of supply chains with simplifications. In our previous works [2], we have first developed a new Petri net model, called Batch Deterministic and Stochastic Petri Nets (BDSPN), for supply chain modeling. With this model, material, information and financial flows of a supply chain can be described in a graphical, concise and integrated way, where operational policies of the chain form one part of the flows. The second step of our study is now to develop a supply chain optimization approach.

For large scale systems such as supply chains, it is very difficult to develop analytical methods for performance evaluation and optimization. As an alternative, many researchers have been seeking for simulation-based optimization

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 732–741, 2007.
© Springer-Verlag Berlin Heidelberg 2007

methods, which use simulation as a performance evaluator [3]. The optimization techniques used in these methods include: metaheuristics such as genetic algorithms, tabu search, simulated annealing and other stochastic optimization methods. By appropriately combining these metaheuristics with simulation, near optimal solutions of a stochastic optimization problem can be obtained in a reasonable computation time. In this paper, we will present a simulation-based optimization method for inventory policies of supply chains that combines a Petri net-based simulation tool with a multi-objective optimization approach - Non dominated Sorting Genetic Algorithms (NSGA-II). In the method, a supply chain is modeled as a batch deterministic and stochastic Petri net, and the performance of the supply chain is evaluated by simulating its Petri net model. A multi-objective optimization approach based on genetic algorithm is used to guide the optimization search process towards global optima. Our approach is tested by using a real-life example - an industrial supply chain of electrical connectors. Numerical results show that the approach can obtain inventory policies significantly better than ones currently used in practice with a reduced cost and an improved service level. The remainder of this paper is organized as follows: Batch deterministic and stochastic Petri nets are briefly introduced in Section 2. The multi-objective genetic algorithm is presented in Section 3. A real-life supply chain is described in Section 4. Our simulation-based optimization method is applied to the optimization of the inventory policies of the supply chain in Section 5 with numerical results and analysis. Several tests are made to evaluate our approach. Concluding remarks are given in Section 6.

2 Batch Deterministic and Stochastic Petri Nets

In our previous works [2], we have proposed and improved a new type of stochastic Petri nets - Batch Deterministic and Stochastic Petri Nets (BDSPN) - to meet the modeling needs of supply chains, where inventory replenishment and distribution operations are usually performed in a batch mode triggered by purchase orders and customer orders, respectively. Batch deterministic and stochastic Petri nets extend Deterministic and Stochastic Petri Nets by introducing batch places and batch tokens. In BDSPN, there are two types of places, discrete places and batch places represented respectively by single circles and squares with an embedded circle. Tokens in a discrete place are viewed indifferently as those in standard Petri nets, while tokens in a batch place, which have sizes, are viewed as different individuals and are represented by Arabic numbers. The tokens of the second type are called batch tokens.

Definition. A batch deterministic and stochastic Petri net is a nine tuple:

$$N = (P, T, I, O, V, W, P, D, \mu_0)$$

where $P = P_d \cup P_b$ is a set of places consisting of the discrete places in set P_d and the batch places in set P_b; $T = T_i \cup T_d \cup T_s$ is a set of transitions consisting of the immediate transitions in set T_i, the deterministic timed transitions in set T_d,

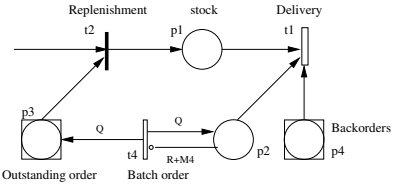


Fig. 1. BDSPN model of a (R, Q) inventory control policy

and the stochastic timed transitions in set T_s; $I \subseteq (P,T)$ and $O \subseteq (T,P)$ define the input arcs and the output arcs of the transitions, respectively, as in standard Petri nets; $V \subseteq (P,T_i)$ defines the inhibitor arcs for immediate transitions with $V \cap I = \emptyset$; W defines the weights for all ordinary arcs and inhibitor arcs. For each arc $a \in I \cup O \cup V$, its weight $W(a)$ is a linear function over the M-marking of the net with integer coefficients. $W(a)$ is assumed to be constant for each arc a associated with a timed transition of $T_d \cup T_s$; $\Pi : T \longrightarrow \aleph$ is a priority function assigning a priority to each transition, where $\aleph$ is the set of nonnegative integers. Timed transitions are assumed to have the lowest priority, i.e., $\Pi(t) = 0$ for all $t \in T_d \cup T_s$, and $\pi(t) \geq 1$ for all $t \in T_i$. $D : T \longrightarrow 0 \cup \Re^+ \cup \Omega$ defines the firing times of all transitions, where $\Re^+$ is the set of positive real numbers, Ω is the set of random variables with a given distribution, $D(t) = 0$ for all $t \in T_i$, $D(t) \in \Re^+$ for all $t \in T_d$, and $D(t) \in \Omega$ for all $t \in T_s$; $\mu_0 : P \longrightarrow \aleph \cup 2^{\aleph}$ is the initial μ-marking, where $2^{\aleph}$ is a superset consisting of all subsets of $\aleph$, $\mu_0(p) \in \aleph$ if $p \in P_d$, and $\mu_0(p) \in 2^{\aleph}$ if $p \in P_b$.

Figure 1 shows the BDSPN model of an inventory system with continue review (R,Q) policy, where R is the reorder point and Q is the fix order quantity. In the model, discrete place p_1 represents the on-hand inventory of the stock and place p_3 represents outstanding orders. Discrete place p_2 represents the on-hand inventory of the stock plus its outstanding orders (the orders that are placed by stock p_1 but not filled yet) The inventory position of the stock equals to $M(p_1)+M(p_3)-M(p_4) = M(p_2)-M(p_4)$. Batch place p_4 represents backorders of the stock (total unfilled customer demand). The operations of the system such as generation of replenishment orders (t_3), inventory replenishment (t_2), and order delivery (t_1) are performed in a batch way because of the batch nature of customer orders recorded in batch place p_4 and the batch nature of the outstanding orders recorded in batch place p_3. The fulfillment of a customer order will decrease on-hand inventory of the stock as well as its inventory level. This is described by the arcs from places p_1, p_4 and p_2 to transition t_1. If a customer order cannot be filled because of stock-out, it will become a backorder (a batch token in place p_4). The continuous inspection of inventory position, i.e., $M(p_2) - M(p_4)$, is represented by immediate transition t_3 and its associated inhibitor arc. When this position is below the reorder point R, i.e., $M(p_2) - M(p_4) < R$, an order with size Q (batch order) will be placed to the supplier (a batch token with the size Q will be created in batch place p_3). If there is a batch token (order) in

place p_3, transition t_2 will be fired after its associated delay, which will replenish stock p_1 by delivering the order to it.

For a more detailed explanation of the BDSPN model (i.e. firing rules and conflicts, temporal policies and behavior) please refer to one of our previous paper [2].

3 Multi-objective Genetic Algorithm

In this section, a simulation-based multi-objective optimization method that combines simulation evaluation of performance and metaheuristic search with a multi-objective genetic algorithm called NSGA-II [6] is adapted for the optimization of inventory policies of supply chains. We have to use multi-objective optimization method [4] because there are more than one objective in supply chain to consider such as the inventory cost and service level. The choice of this metaheuristic is motivated by its potential as a stochastic search method, the facility of encoding inventory policy parameters of supply chains and the ability to find multiple Pareto optimal solutions in one single simulation run. Genetic algorithms are very well adapted to treat multi-objective optimization problems. Indeed, in recent years, extremely robust methods are proposed [1,5] to solve multi-objective optimization problems using genetic algorithms. According to the studies which have been conducted on several multi-objectives algorithms, the NSGA-II is classified among the best and the most popular algorithms in term of convergence and diversity of solutions.

3.1 NSGA II Principle

NSGA-II computes successive generations on a population of solutions belonging non- dominated fronts. The non-dominated set is identified and constitutes the non-dominated front of level 1 or front 1. In order to find the individuals in the next non-dominated front, the solutions of front 1 are discounted temporarily and the above procedure is repeated. This process continues until all fronts are identified. In order to maintain diversity in the population, the crowding-distance is used. The overall structure of the NSGAII is specified by the algorithm 1.

3.2 GA Components for Bi-objective Supply Chain Optimization

In this study, we consider a supply chain composed of n interrelated stocks. Each stock uses a batch ordering policy (R, Q) for its inventory replenishment. The supply chain has two objectives, namely total inventory cost and service level, to optimize. Basic operations that characterize our adapted NSGA-II are explained as follows.

Encoding. For batch ordering policies, only two parameters are involved: the reorder point R and the order quantity Q. For a supply chain with n stocks controlled by batch ordering policies, each chromosome is encoded as $[R_1, Q_1, \ldots,$

$R_i, Q_i, \ldots, R_n, Q_n]$, where R_i and Q_i are the reorder point and the order quantity of the batch ordering policy of the i_{th} stock, respectively, $i = 1, 2, \ldots, n$. For a supply chain with other types of inventory policies or a mixture of multiple types of policies, the chromosome can be similarly encoded. The length of the chromosome is the total number of the parameters of all inventory policies of the supply chain. In the following, we assume that the order quantity Q_i for each stock i is given. Therefore, (Q_i) are constant, so each chromosome is encoded as $[R_1, R_2, \ldots, R_n]$. In this sequence of R_i, each one is encoded in binary.

Algorithm 1: NSGAII overall structure

```
Create the initial population P of size n
Evaluate the n solutions using simulation
Sort P by non domination
Computed the crowding distance of each solution
repeat
  Create and add n children at the end of P (with genetic operators: selection,
  crossover and mutation of two parents)
  Sort P by non domination
  Computed the crowding distance of each solution
  newP ← ∅
  i ← 1
  while |newP| + |front(i)| ≤ n do
    Add front(i) to newP
    i ← i + 1
  end while
  missing ← n − |newP|
  if missing ≠ 0 then
    Sort the solutions by descending order of the crowding distance
    for j ← 1 to missing do
      Add the jth solution of front(i) to newP
    end for
    P ← newP
  end if
until Stopping Criterion
```

Initial Population. The NSGA-II starts the search by generating a population of candidate solutions. In our implementation, this population is randomly generated according to uniform distributions. That is, the parameters (gene values) R_i are randomly generated according to uniform distributions $U[R_{imin}, R_{imin}]$ where R_{imin} and R_{imax} are the minimum and the maximum possible values of R_i.

Chromosomes Evaluation and Selection. In our study, each chromosome is evaluated through simulation. The simulation is controlled by two control parameters: the number of simulations per chromosome N and the length of each simulation run T. For each chromosome the fitness value f_k is evaluated on its average value over the N simulations.

Selection is a process in which chromosomes are copied according to their fitness function value or their rank value. In this study, the tournament parent

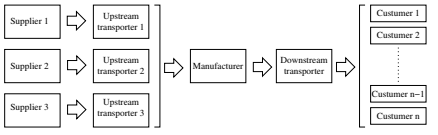


Fig. 2. Structure of an industrial supply chain

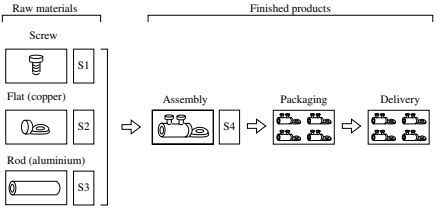


Fig. 3. Manufacturing process of an electrical connector

selection is used. Tournament selection is one of many selection methods in genetic algorithms which runs a *tournament* among a few individuals chosen at random from the population and selects the winner (the one with the best fitness) for crossover. So, the chance of each chromosome being selected is directly proportional to its fitness value.

Crossover and Mutation. The crossover produces new offspring chromosomes from parent individuals. Two new chromosomes are created by exchanging some genes of two parent chromosomes. We use in our implementation single-point crossover. This kind of crossover creates a pair of offspring by exchanging parts of the parents after a crossover point. The crossover point is usually a randomly selected integer.

The mutation introduces some extra variability into the current population. The function of mutation is to maintain the diversity of the population in order to prevent too fast convergence of the algorithm. The probability of mutation is $1/l$ where l is the string length of our binary-coded variables.

Stopping Conditions. There are no universally stopping conditions accepted for multi-objective genetic algorithms. In this study, we simply stop our algorithm after a given number of iterations (Ng).

4 Industrial Case

In this section, a real life supply chain is presented. For confidential reason, the name of the company is not mentioned. The supply chain is composed of three suppliers, three upstream transporters with one for each supplier, a manufacturer, a downstream transporter for the manufacturer and a set of customers (see Figure 2). The manufacturer produces an electrical connector for high voltage lines. There are three types of flows in the supply chain: material flows, information flows and financial flows. For material flows (see Figure 3), the manufacturer needs three raw materials for the production of the connector: flats, rods, and screws. They are purchased from suppliers 1, 2 and 3, respectively. These materials are delivered to the manufacturer by the corresponding upstream transporters. At the manufacturer site, rods of aluminium are cut into shafts with a pre-specified length, flats are bored and ground, and the finished

product is then produced by assembling a shaft, a flat, and two screws. The product will be further packaged and delivered to customers by the downstream transporter. For information flows, the manufacturer receives customer demand in the form of orders. When the inventory position of the finished product (Stock 4) is below a pre-specified reorder level, an assembly order with a given batch size will be released to the assembly line of the manufacturer. For the raw material stocks (S1, S2 and S3), when their inventory positions are below their reorder levels, a purchase order with a given quantity will be placed to their corresponding suppliers. Each purchase or assembly order contains the information such as order release time and order quantity, which are determined by an inventory policy involved. The inventory policies of all the stocks are periodic review batch ordering policy. For financial flows, customers pay the manufacturer within a given time period after receiving their ordered finished products. The manufacturer pays its suppliers similarly. The suppliers and the manufacturer pay their transporters within a given time period after the delivery of raw materials or finished products.

5 Computational Experiments

In this study, we have programmed our adapted NSGA-II algorithm on LINUX station using C^{++} under KDevelop environment. The algorithm is tested on real data of an industrial case.

5.1 Parameters Setting

Our algorithm was run with different values of generation number (from 10 to 1000), with an initial population size of 100 and with the following parameters referring to the NSGA-II [6]: the probability of crossover is 0.9 and the probability of mutation is $0.02128(= 1/47)$. Also the genetic operations such as the Pareto dominance ranking procedure and the elitist selection are used.

5.2 Computational Results

With the BDSPN model, the performance of the real-life supply chain is evaluated by simulation using a BDSPN simulator (a C^{++} program) developed by us. In the model, the mean replenishment lead times for stocks S1, S2, S3, S4 are 40, 20, 70, and 20 days, respectively. The mean transportation lead times for all transporters are 2 days. The standard deviations of these lead times are negligible. Customer orders arrive randomly with the inter-arrival times subject to an exponential distribution with mean value 0.0355. The reorder points of stocks S1, S2, S3, S4 are 2300, 590, 2000, 400, respectively, while the order quantities of these stocks are 5000, 3000, 9300, 2000, respectively. The annual holding costs of raw materials and final products in stocks S1, S2, S3, S4 are

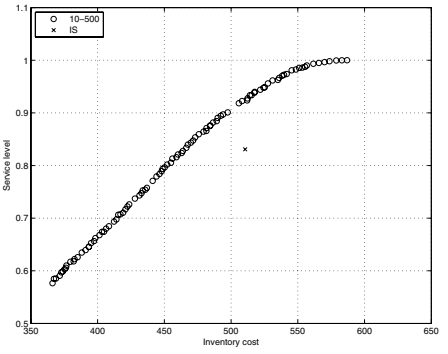


Fig. 4. The distribution of Pareto optimal front F_6 (10-500) and the IS point

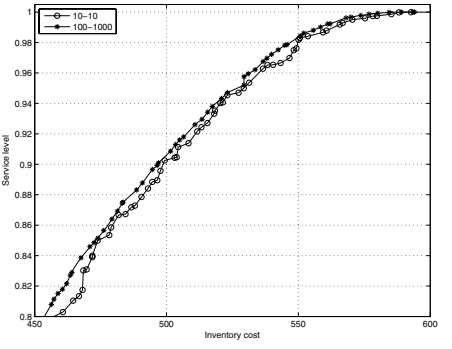


Fig. 5. Comparison between front F_1 (10-10) and front F_{16} (100-1000)

12% of their prices which are 0.3€, 0.6€, 0.16€, and 3€, respectively. Note that the considered performance criteria of the supply chain include average inventory level and service level of each stock, where the service level is defined as the probability that customer orders can be filled on time. The first criterion is easy to obtain since it corresponds to the average number of tokens in the discrete place representing the on-hand inventory of a stock. For the evaluation of the service level, it needs to obtain the total time that the discrete place has no token while the corresponding batch place that represents customer orders is not empty in each simulation run. This can be done by observing the markings of the two places during the simulation. Because of the stochastic nature of the model, the simulation has to be replicated for many times with a long time horizon (simulation length) to get reliable estimates of the performance indices. Each index is evaluated as its average value over all simulation replications. The accuracy of the evaluation depends on the number of simulations performed and the simulation length adopted.

According to the situation of the manufacturer, the simulation horizon is set as 3 years with a warm up period of 3 months, in order to reach a steady state. For each set of decision variables, 10 and 100 replications are simulated. The initial performances using the real data of the manufacturer and obtained by simulation are **510.56** for the mean total inventory cost per day, which is the sum of the mean inventory costs of the four stocks, and the mean service level to the final customers are **83.08 %**. The goal of the manufacturer is to maximise the service level to the final customer and to minimize the mean total inventory cost. To reach this goal, the simulation-based optimization approach is applied to optimize the inventory policies of the supply chain.

Evaluation Criteria. In the study of a multi-objective optimization problem based on genetic algorithms, the central issue is to compare two fronts obtained by resolving this problem with different parameters of the algorithm. This

Table 1. Computational results

Na	N	Ng	Time(s)	$\|F\|$	$C(F_i;F_1)$	$C(F_1;F_i)$	μ	$\bar{\mu}$
F_1	10	10	71	100	0	0	0.00	0.00
F_2	10	25	180	100	0	84	0.98	0.0098
F_3	10	50	335	100	0	89	1.08	0.0108
F_4	10	100	665	100	0	94	1.15	0.0115
F_5	10	250	1343	100	0	92	1.14	0.0114
F_6	10	500	3292	100	0	89	1.10	0.0110
F_7	10	750	4926	100	0	92	1.14	0.0114
F_8	10	1000	6542	100	0	95	1.21	0.0121
F_9	100	10	727	100	25	31	0.30	0.003
F_{10}	100	25	1713	100	20	67	0.72	0.0072
F_{11}	100	50	3387	100	0	78	0.93	0.0093
F_{12}	100	100	6624	100	1	90	0.99	0.0099
F_{13}	100	250	13302	100	0	88	1.01	0.0101
F_{14}	100	500	32669	100	2	85	1.04	0.0104
F_{15}	100	750	49077	100	0	85	1.02	0.0102
F_{16}	100	1000	65342	100	0	86	1.00	0.0100
IS	–	–	1	1	100	0	–	-0.09

comparison is important because it allows choosing the efficient algorithm parameters. In this study, two evaluation criteria are used:

- μ Distance: This measure is proposed by Riise [7]. $\mu = \sum_{i=1}^{N} d_i$, where d_i is the distance between a solution $i \in F_1$ and its orthogonal projection on F_2. The μ value is negative if F_1 is below F_2, and positive otherwise.
  Since μ depends on the number of solutions in $F_1(N)$, a normalized measure is generally taken: $\bar{\mu} = \frac{\mu}{N}$
- Zitzler Measure ($C(F_1;F_2)$): This measure has been proposed by Zitzler [9]. It represents the percentage of solutions in F_1 dominated by at least one solution of F_2. Since this measure is not symmetrical, it is necessary to calculate $C(F_2;F_1)$. Therefore, F_1 is better than F_2 if $C(F_1;F_2) < C(F_2;F_1)$.

Results. To test our optimization method, several sets of parameters are used and the corresponding results are presented in table 1. Na, N and NG are respectively the name of the parameter set, the number of simulations per individual and the number of generations. F_i is the optimal Pareto front and IS is the industrial solution described before with the performances (510.56€, 83.08%). The number of solutions $|F|$ in all Pareto optimal fronts F_i is always equal to 100. Each optimal Pareto front F_i and the solution IS are compared to the reference front F_1 on two evaluation criteria, the Zitzler measures $C(F_i;F_1)$ and $C(F_1;F_i)$ and the μ distance. Results obtained show the results obtained with all parameters settings are better than the industrial solution IS but with a longer computational time (more than 18 hours for F_{16}). The longer computational time is acceptable since the optimization of the inventory policies can be done off time. Graphically, Figure 4 gives an example of the optimal Pareto distribution of the front F_6. Each point represents a specific solution that is Pareto-optimal. We can also discover that the points are evenly distributed along the front. The IS point is below the optimal Pareto front. The figure 5 compares two optimal Pareto fronts F_1 and F_{16}.

6 Conclusion

In this paper, we have developed a simulation-based method for the optimization of the inventory policies of supply chains with multiple objectives. This method combines simulation evaluation of performance with a genetic algorithm that guides an optimization search process toward global optima. The simulation evaluation is based on a batch deterministic and stochastic Petri net modeling tool we developed. Our method is tested by a case study that optimizes inventory policies for a industry supply chain. Our NSGA2-based approach is performed with several different parameters. The different solutions represented by Pareto fronts are compared with two evaluation criteria : μ-distance and Zitzler measure. Numerical results show that our approach can obtain inventory policies better than the ones currently used in practice with a reduced inventory cost and an improved service level. A global optimization approach of the inventory policies or a robustness analysis applied to the parameters of the approach could be interesting for future research.

References

1. E.K. Burke and J.D. Landa Silva. The influence of the fitness evaluation method on the performance of multiobjective search algorithms. *European Journal of Operational Research*, 169(3):875–897, 2006.
2. H. Chen, L. Amodeo, F. Chu, and K. Labadi. Modelling and performance evaluation of supply chains using batch deterministic and stochastic petri nets. *IEEE Transaction on Automation Science and Engineering*, 2(2):132–144, 2005.
3. C.A. Coello Coello. An updated survey of ga-based multiobjective optimization techniques. *ACM Computing Surveys*, 32(2):109–143, 2000.
4. Y. Collette and P. Siarry. *Optimisation multi-objectif*. Eyrolles, 2002.
5. J.S.R. Daniel and C. Rajendran. Heuristic approaches to determine base-stock levels in a serial supply chain with a single objective and with multiple objectives. *European Journal of Operational Research*, 2005.
6. K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002.
7. A. Riise. Comparing genetic algorithms and tabu search for multiobjective optimization. In *Conference proceedings IFORS*, Edinburgh, UK., 2002.
8. S. Tayur, R. Ganeshan, and M. Magazine. *Quantitative Models for Supply Chain Management*. Kluwer Academic Publishers, 1998.
9. E. Zitzler and L. Thiele. Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. *IEEE Transactions on evolutionary computation*, 3(4):257–271, 1999.

Scheduling a Fuzzy Flowshop Problem with Flexible Due Dates Using Ant Colony Optimization

Sezgin Kilic

Department of Industrial Engineering, Air Force Academy,
34149, Yesilyurt, Istanbul, Turkey
{Sezgin.Kilic,s.kilic}@hho.edu.tr

Abstract. Most of the work about flowshop scheduling problems assume that the problem data are known exactly at the advance or the common approach to the treatment of the uncertainties in the problem is use of probabilistic models. However, the evaluation and optimization of the probabilistic model is computationally expensive and rational only when the descriptions of the uncertain parameters are available from the historical data. In addition, a certain amount of delay on due dates may be tolerated in most real-world situations although they are handled as crisp dates in most of the previous papers. In this paper we deal with a flowshop scheduling problem with fuzzy processing times and flexible due dates. Schedules are generated by a proposed algorithm in the context of ant colony optimization metaheuristic approach.

Keywords: fuzzy processing time, fuzzy due date, flow shop scheduling, ant colony optimization, necessity measure.

1 Introduction

The flowshop scheduling problems are made up of *n* similar jobs which have the same order of processing on *m* machines. The problem is NP-hard, only some special cases can be solved efficiently [1]. Even though, the flowshop scheduling problems has often been investigated, very little of this research is concerned with the imprecise or ambiguous impressions in problem variables such as processing times and due dates. Weaknesses of modelling have been the major obstacle that prevents traditional techniques from being used in industrial applications with amazing speed increase of modern processors and new techniques of resolution such as metaheuristics [2]. In order to describe and characterize real-world problems closely it is more suitable to consider fuzzy processing time and fuzzy due date particularly in human incorporated systems since on every occasion of human interface there may be some deviation from the deterministic value and there will be difficulties in applying and validating the generated schedule. Concerning due date, solution to a problem which is modeled with crisp due dates will be invalid for any amount of delay on due dates. However, in real-world applications it is possible to see that a certain amount of delay may be tolerated. There may be situations where a schedule is also valid for same delays with less single customer satisfaction but greater acquisitions in total manner as a consequence of expanded solution space through flexible due dates.

M. Giacobini et al. (Eds.): EvoWorkshops 2007, LNCS 4448, pp. 742–751, 2007.
© Springer-Verlag Berlin Heidelberg 2007

In this paper we used ant colony optimization (ACO) approach to generate a schedule for the cases where the processing times are uncertain and due dates are flexible in some range. Flexible due dates are handled as fuzzy due dates. The problem is denoted as fuzzy permutation flowshop (FPFS) problem. It is assumed that the planner is able to approximate the imprecise data using fuzzy sets. Initially we explain how to calculate the completion time of each job using fuzzy arithmetic. Then we focused on the scheduling criteria in order to decide which schedule is the best. We used the necessity measure in order to define a certainty level of fuzzy completion time of a job to be earlier than the flexible due date.

There is some research on FPFS problem but to our knowledge there have not been any work on FPFS using an ant colony optimization approach. The first application of fuzzy set theory on a flowshop problem as means of analyzing performance characteristics was by [3] in 1992. A multi-objective genetic algorithm for large size flowshop scheduling problems with fuzzy processing time is proposed in [4]. In [5] authors applied a fuzzy approach to the treatment of processing time uncertainty in flowshop plants and new product development process scheduling problems.

The remainder of this paper is organized as follows. In the next Section, we introduce the FPFS problem. Then we introduce the setting of the ACO algorithm in Section 3. The computational experiments are presented in Section 4. Finally we conclude the paper with a summary in Section 5.

2 Permutation Flowshop Scheduling Problem with Fuzzy Processing Time and Fuzzy Due Date

In general a *n x m* permutation flowshop scheduling problem is formulated as follows. Let *n* jobs (j=1,2,…,n) be processed on *m* machines (*k*=1,2,…,*m*) in the same order. Only permutation schedules are considered, where the order in which each machine processes the jobs is identical for all machines. Hence a schedule is uniquely represented by a permutation of jobs. The processing of each job on each machine is an operation which requires the exclusive use of the machine for an uninterrupted duration called the processing time. Let the processing time of job *j* on machine *k* be t_{jk}. The objective is to find a schedule that minimizes the makespan. In this paper a permutation flowshop scheduling problem with fuzzy processing times and fuzzy due dates is formulated as a fuzzy permutation flowshop scheduling (FPFS) problem.

Using fuzzy numbers for representing the uncertainty in processing times is very plausible for real world applications. If a decision maker estimates the processing time of the job *j* on machine *k* (t_{jk}) as an interval rather than a crisp value then the interval can be represented as a fuzzy number. In this paper, the fuzzy due date, $\tilde{D}_j$ is represented by the degree of satisfaction with respect to completion time of job *j* and denoted by a double (d_j^1, d_j^2) as shown in Fig. 1(a). d_j^2's are generated by adding acceptable delays on crisp due dates. The fuzzy processing time of job *j* on machine *k* is represented by a triangular fuzzy number (TFN) $\tilde{T}_{jk}$ and denoted by a triplet ($a_{jk}^1, a_{jk}^2, a_{jk}^3$) as illustrated in Fig.1(b).

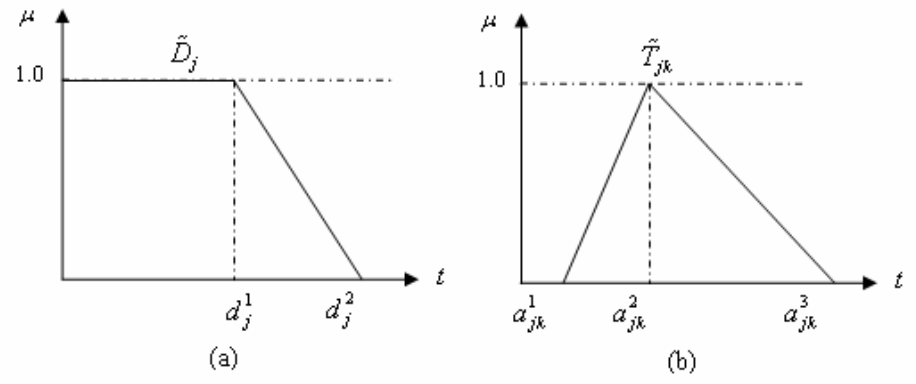


Fig. 1. Fuzzy due date and fuzzy processing time

2.1 Fuzzy Operations for Generating a Schedule

The addition operator and the maximum operator are required for calculating fuzzy completion time of a schedule. The addition operator does not distort the shape of fuzzy numbers. The sum of two triangular fuzzy numbers is also a triangular fuzzy number. Conversely, the maximum operator can disfigure the triangular shape of fuzzy numbers. In Fig. 2(a) two fuzzy numbers $\tilde{A}$ and $\tilde{B}$ are represented. Fig. 2(b) represents the $max\{\tilde{A}, \tilde{B}\}$.

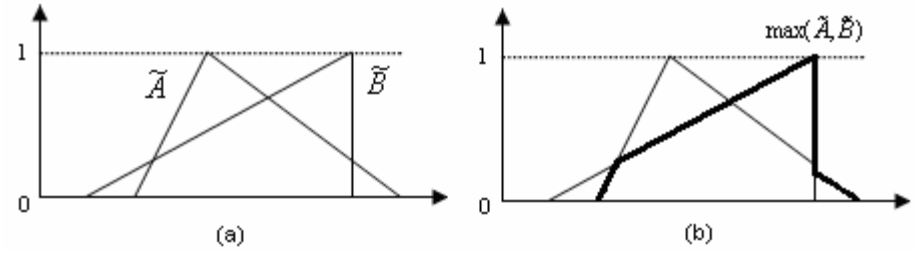


Fig. 2. Fuzzy maximum operation [6]

Triangular fuzzy numbers can be represented by triplets without any information lost but when the triangularity is distorted we need a more complex definition to represent a fuzzy number.

A fuzzy set $\tilde{A}$ of the universe *X* is specified by a membership function $\mu_{\tilde{A}}(x)$, which takes its value in the interval [0,1]. For each element *x* of *X*, the quantity $\mu_{\tilde{A}}(x)$ specifies the degree to which *x* belongs in $\tilde{A}$. $\tilde{A}$ is completely characterized by the set of ordered pairs:

$$\tilde{A} = \left\{ \left(x, \mu_{\tilde{A}}(x) \right) \middle| x \in X \right\} . \quad (1)$$

The α-level set (α-cut) of a fuzzy set $\tilde{A}$ is a crisp subset of *X* given by

$$\tilde{A}_\alpha = \left\{ x \in X \,\middle|\, \mu_{\tilde{A}}(x) \geq \alpha \right\} \qquad \forall \alpha \in (0,1] \,. \tag{2}$$

and it can be represented by $\tilde{A}_\alpha = \left[x_\alpha^L, x_\alpha^R \right]$ where x_α^L and x_α^R are crisp values denoting the x values where the $\mu_{\tilde{A}}(x) = \alpha$ for left and right sides of the $\tilde{A}$ respectively. Calculating maximum of two fuzzy numbers requires infinite computations for every $\alpha \in (0,1]$ using Eq. 3.

$$\max\left\{ \tilde{A}_\alpha, \tilde{B}_\alpha \right\} = \left[\max(a_\alpha^L, b_\alpha^L), \max(a_\alpha^R, b_\alpha^R) \right] \quad \forall \alpha \in (0,1] \,. \tag{3}$$

Besides, if the fuzzy numbers are not triangular the addition operator also requires infinite computations for every $\alpha \in (0,1]$ using Eq. 4. Let, $\tilde{A}_\alpha = \left[x_\alpha^L, x_\alpha^R \right]$ and $\tilde{B}_\alpha = \left[y_\alpha^L, y_\alpha^R \right]$;

$$\tilde{A} + \tilde{B} = \left[x_\alpha^L + y_\alpha^L, x_\alpha^R + y_\alpha^R \right] \qquad \forall \alpha \in (0,1] \,. \tag{4}$$

It is impossible to make all computations for every $\alpha \in (0,1]$. However, good approximations can be obtained by performing these computations at specific values of α, rather than all values. The computations for addition and maximum operations are approximately done by interval arithmetic for 21 level sets ([alpha] cuts with α=0.00, 0.005, 0.10, … , 1.00) in this paper. Let; $\pi = \{\pi(1), ..., \pi(j), ..., \pi(n)\}$ be a permutation of jobs (a candidate solution); $\tilde{T}_{\pi(j),k}$ be the fuzzy processing time of job $\pi(j)$ on machine *k*; $\tilde{C}_{\pi(j)}$ be the fuzzy completion time of job $\pi(j)$; and $\tilde{c}_{\pi(j),k}$ be the fuzzy completion time of job $\pi(j)$ on machine *k*. We can find fuzzy start and completion times for each job on each machine for a permutation π as follows;

$$\tilde{c}_{\pi(j),k} = \max(\tilde{c}_{\pi(j-1),k}, \tilde{c}_{\pi(j),k-1}) + \tilde{T}_{\pi(j),k} \qquad \forall j > 1, \forall k > 1 \,. \tag{5}$$

$$\tilde{C}_{\pi(1),k} = \tilde{C}_{\pi(1),k-1} + \tilde{T}_{\pi(1),k} \qquad \forall k > 1 \,. \tag{6}$$

$$\tilde{C}_{\pi(j),1} = \tilde{C}_{\pi(j-1),1} + \tilde{T}_{\pi(j),1} \qquad \forall j > 1 \,. \tag{7}$$

$$\tilde{c}_{\pi(1),1} = \tilde{T}_{\pi(1),1} \,. \tag{8}$$

The FPFS problem is considered with the objective of minimizing the makespan. We seek for a schedule with minimum makespan, furthermore we want to be sure as much as possible for any job to be finished not later than its unacceptable due date

time (d_j^2). We take advantage of necessity measure for the degree of certainty about the schedule to be in boundaries of the due dates. Let us shortly recall the necessity measure $N(P)$: it describes to what extend we have to believe in $x \in P$, or how much we can doubt about $x \notin P$. If P ($P = -\infty, d_j^2[$) describes the fuzzy set of time values earlier than (necessarily less than) d_j^2 , and $]\tilde{C}_j, +\infty)$ describes the fuzzy set of numbers necessarily greater than $\tilde{C}_j$ then;

$$\mu_{]\tilde{C}_j,+\infty)}(d_j^2) = N_{\tilde{C}_j}(-\infty, d_j^2[) = \inf_{u \geq d_j^2}(1 - \mu_{\tilde{C}_j}(u)) \ . \tag{9}$$

is interpreted as the certainty that "by the moment d_j^2 processing of job *j* has already been finished on the last machine".

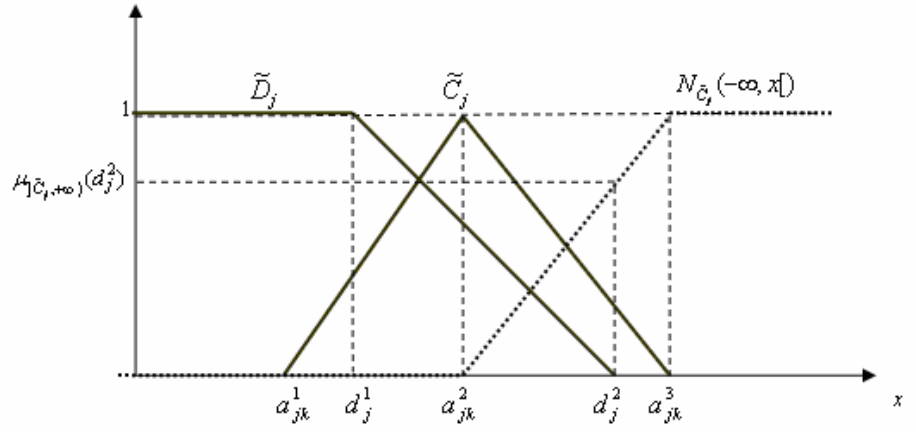


Fig. 3. Necessity measure for d_j^2 to be greater than $\tilde{C}_j$

Necessity measure of a schedule (Nec(π)) is the minimum necessity measure of a job in the schedule (Eq. 10);

$$Nec(\pi) = \min(N_{\tilde{C}_j}(-\infty, d_j^2[)) \quad \forall j \ . \tag{10}$$

A schedule is regarded as acceptable if Nec(π)>0.

2.2 Comparison of Fuzzy Makespans

When fuzzy data are incorporated in to the scheduling problem the measure of the makespan values of the alternative schedules are no longer crisp numbers, they are fuzzy numbers. Since a fuzzy number represents many possible real numbers that have different membership values, it is not easy to compare the makespans to

determine which schedule is favored. There exists a large body of literature that deals with the comparison of fuzzy numbers. More recently, [7] have proposed the "area compensation (AC)" method for comparing fuzzy numbers based on compensation of the areas determined by the membership functions. They have shown that AC is a robust ranking technique which has compensation, linearity and additivity properties, compared to other ranking techniques. In addition, AC yields results consistent with the human intuition. The AC of a fuzzy number $\tilde{\tilde{X}} = \left[x_\alpha^L, x_\alpha^R\right]$ is defined by,

$$AC(\tilde{X}) = 0.5\int_0^1 (x_\alpha^L + x_\alpha^R)d\alpha \ . \tag{11}$$

A schedule with fuzzy makespan $\tilde{\tilde{X}}_1$ will be preferred over a schedule with fuzzy makespan $\tilde{\tilde{X}}_2$ if $AC(\tilde{\tilde{X}}_1) < AC(\tilde{\tilde{X}}_2)$.

3 Proposed ACO Algorithm

3.1 Ant Colony Optimization Approach

Ant Colony Optimization (ACO) is a population based, cooperative search metaphor inspired by the foraging behavior of real ants. In ACO algorithms, artificial ants collectively search for good quality solutions to the optimization problem [8]. Common structure of the ACO algorithms can be illustrated as follows;

Step 1. Initialize the pheromone trails and parameters.
Step 2. While (termination condition is not met) do the following:
Construct a solution,
Improve the solution by local search,
Update the pheromone trail or trail intensities.
Step 3. Return the best solution found.

3.2 Description of the Proposed ACO Algorithm

In this paper we proposed an algorithm based on MAX-MIN Ant System (MMAS) [9]. We made modifications and proposed extensions described below in order to schedule a FPFS problem.

3.2.1 Initializing the Pheromone Trails and Parameters

One of the main differences of MMAS from other ACO algorithms is that it limits the possible range of pheromone trail values to the interval $[\tau_{\min}, \tau_{\max}]$ in order to avoid search stagnation. $\tau_{\max} = 1/Z_{\text{glb}}^{best}$ and $\tau_{\min} = \tau_{\max}/a$, where Z_{glb}^{best} is the makespan of the best solution found up to now and a is a parameter. $\tau_{\max}$ and $\tau_{\min}$ are updated each time when a new Z_{glb}^{best} is found.

τ_{ip} denotes the quantity of trail substance for job *i* on p_{th} position. It gives the degree of desirability for an ant to choose job *i* for position *p* for the schedule being generated by itself. Initial values for τ_{ip} are set to $\tau_{\max}$.

3.2.2 Construction of a Solution by an Artificial Ant

In order to schedule the FPFS problem, each ant starts with a null sequence and makes use of trail intensities for selecting a job for the first position, followed by the choice of an unscheduled job for the second position, and so on. Each selection is made with the application of a probabilistic choice rule, called random proportional rule, to decide which job to select for the next position. In particular, the probability with which the ant will choose job *i* for the p_{th} position of its schedule is denoted as P_{ip};

$$P_{ip} = \begin{cases} \dfrac{(\tau_{ip})^{\alpha} \times (1/d_i^2)^{\beta}}{\sum_{l \in \vartheta} (\tau_{lp})^{\alpha} \times (1/d_i^2)^{\beta}} & \text{if } i \in \vartheta \\ 0 & \text{if } i \notin \vartheta \end{cases} \tag{12}$$

ϑ is the set of unscheduled jobs by the ant until now. α and β are two parameters which determine the relative influence of pheromone trail and the heuristic information. At the early stages of the algorithm d_i^2 values will direct the search to find acceptable solutions but as the search continues trail intensities will have more effect on generating probabilities and the algorithm will be searching for the best solution.

3.2.3 Updating Trail Intensities

After a complete sequence is constructed by each ant and possibly improved by the insertion-move, the trails are updated. In MMAS only one of the ants is allowed to add pheromone. The ant which generated the best tour in the current iteration or the ant which have generated the best tour since the start time of the algorithm is chosen for trail updating. Let Z_{iter}^{best} denotes the best makespan found by the ants in current iteration. The ant which generated the best schedule in the current iteration or the ant which have been generated the best-so-far schedule updates the trails as follows,

$$\tau_{ip}^{new} = \begin{cases} (1-\rho).\tau_{ip} + \left\{ \dfrac{1}{Z_{iter}^{best}}, \dfrac{1}{Z_{\text{glb}}^{best}} \right\} \\ (1-\rho).\tau_{ip} \end{cases} \tag{13}$$

where ρ denotes the pheromone evaporation rate $(0<\rho<1)$. The parameter ρ is used to avoid unlimited accumulation of the pheromone trails and it enables the algorithm "forget" bad decisions previously taken. In the next iteration, $\tau_{ip} = \tau_{ip}^{new}$ will be used in Eq. (13) for selecting jobs for positions.

4 Computational Experiments

The proposed algorithm was coded in Matlab 7.0 R14 and run on an Intel Pentium III processor 801 MHz and 256 MB of RAM PC. A problem with eight jobs and four machines from [10] is modified by adding due dates is handled in order to inspect the search process. Parameter values are used as follows; number of ants=10, α=1, β=2, ρ=0.05, a =5 and 30% of the ants make global trail updating. Values of the parameters are determined in accordance with experiences gained in the numerical experimentations and the recommendations based on the previous studies. 21-point approximation is used for fuzzy addition and maximum operations. AC is used for fuzzy ranking. Table 1 presents some of the schedules generated during a search for 1000 cycles.

Table 1. Some of the schedules generated for the problem

Job Sequence (π)	AC	Nec(π)	Makespan (TFN)	Feature
2,5,6,8,3,7,1,4	189.06	0.158	(137,178,268)	Best makespan with positive neccessity
2,8,3,6,1,4,7,5	195.01	0.819	(133,186, 280)	Maximum neccessity
No solution	-	-	-	Best makespan when due date is crisp ($d_j^2 = d_j^1$) with positive necessity
2,8,3,4,1,6,7,5	204.56	0.765	(143,198,283)	In ascending order of due dates
2,5,3,6,8,7,1,4	187.43	negative	(134,177,270)	Best makespan while necessity may be negative

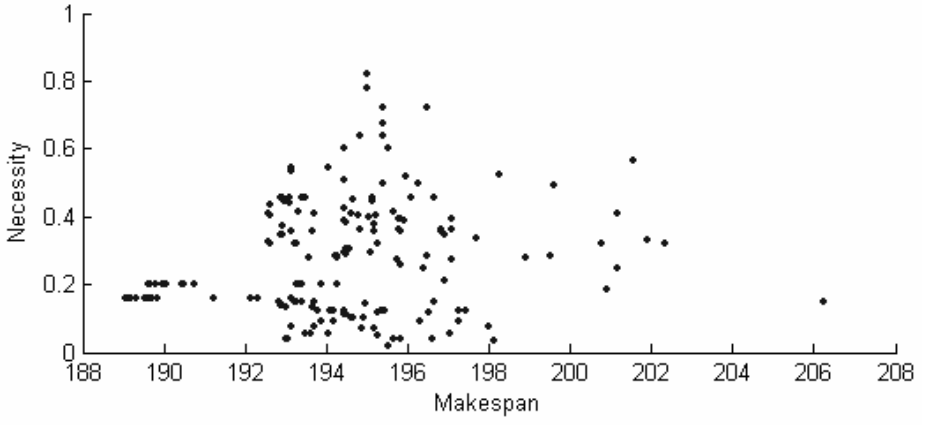


Fig. 4. Makespan values (AC) and corresponding necessity measures for generated schedules

As seen in Table 1 best schedule has a AC of 189.06 with Nec(π)=0.186, whereas a schedule with a Nec(π)= 0.819 has a AC value of 195.01 which is 4.4 times more certain to be finished before due dates but 3% longer in makespan. If the flexibility in due dates were omitted for the problem we could not have generated a feasible schedule with positive necessity measure as seen in last row of Table 1.

Fig. 4 represents the AC values and necessity measures of all the schedules generated. In 592 of 1000 cycle, algorithm generated schedules with positive necessity measure. It can be seen that some schedules with same AC value have different necessity measures or some schedules with same necessity measure have same AC values.

As illustrated in Fig. 4, flexible due dates allow scheduler to make decision between alternative schedules.

5 Conclusion

Flowshop scheduling is often stated as a deterministic problem. However, in most practical applications some of the data may be imprecise and there may be flexibilities for some constraints. In this paper we tried to handle the imprecision of data and flexibility in constraints by fuzzy sets. The flowshop problem is NP-hard in crisp case and the complexity of the problem seriously increases when it is fuzzified. Therefore we used an ant colony optimization algorithm in order to generate good solutions in reasonable time periods. Computational experiments illustrate that, even for a small sized problem, accepting some delays for crisp due dates expands the solution space and better solutions may be accessible. In addition a robust solution model is handled by using fuzzy durations and flexible due dates which represent ambiguity and tolerability respectively. It should be noted that there are several possible extensions to this study in the future research, for instance FPFS problem can be formulated as a multiple objective decision making with makespan and certainty objectives.

References

1. Rinnooy-Kan A.H.G.: Machine scheduling problems: Classification, complexity and computations. Hagues: Martinus Nijhoff (1976)
2. Fortemps, P. : Introducing flexibility in scheduling: the preference approach. In: Slowinski, R., Hapke M. (eds.): Scheduling Under Fuzziness. Physica Verlag, Heidelberg New York (2000) 61-79
3. McCahon C.S., Lee E.S.: Fuzzy job sequencing for a flow shop. European Journal of Operational Research 62 (1992) 294-301
4. Ishibuchi H., Murata T., Lee K.H.: Formulation of fuzzy flowshop scheduling problems with fuzzy processing time. In: Proceedings of the 15th IEEE Int. Conf. on Fuzzy Systems. New Orleans (1996)
5. Balasubramanian J., Grossmann I.E.: Scheduling optimization under uncertainty – an alternative approach. Computers and Chemical Engineering 27 (2003) 469-490
6. Dubois, D., Fargier, H., Fortemps, P.: Fuzzy scheduling : Modelling flexible constraints vs. coping with incomplete knowledge. European Journal of Operational Research 147 (2003) 231-252

7. Fortemps P., Roubens M.: Ranking and defuzzification methods based on area compensation. Fuzzy Sets and Systems 82 (1996) 319-330
8. Dorigo M., Stützle T.: Ant Colony Optimization. The MIT Press, Massachusetts, London (2004)
9. Stützle T., Hoos H.H.: MAX-MIN Ant System. Future Generation Computer Systems 16 (2000) 889-914
10. Balasubramanian, J., Grossmann, I.E.: Scheduling optimization under uncertainty – an alternative approach. Computers and Chemical Engineering 27 (2003), 469-490

Author Index

Aggarwal, Varun 21
Alba, Enrique 101
Alfaro-Cid, Eva 129, 169
Alharbi, Abir 657
Almejalli, Khaled 688
Amodeo, Lionel 732
Ando, Daichi 577
Assayag, Gérard 488
Aurnhammer, Melanie 351
Aydin, Mehmet E. 111

Bäck, Thomas 391
Berry, Marsha 498
Bhanu, Bir 291
Bilotta, Eleonora 585
Bisogno, Fabio 431
Bovenkamp, Ernst G.P. 391
Brabazon, Anthony 189

Caetano, Marcelo 477
Cagnoni, Stefano 241
Carlesi, Maria 233
Carpentier, Grégoire 488
Castillo, Pedro Angel 129, 712
Chen, Haoxun 732
Choi, Jeong-Yoon 593
Ciesielski, Vic 498
Cinel, Caterina 301
Citi, Luca 301
Cordón, Oscar 415
Cortez, Paulo 71
Costa, Ernesto 617
Cruz-Cortés, Nareli 330, 359
Cupellini, Enrico 585

D'Souza, Daryl 498
Dag, Hasan 218
Dahal, Keshav 688
Dahlsted, Palle 577
Damas, Sergio 415
Davis, Tom 508
De Falco, Ivanoe 251
de la Fraga, Luis Gerardo 330, 359
del Rey, Martín Angel 52
Della Cioppa, Antonio 251
Dijkstra, Jouke 391
Dimitriou, Loukas 668, 678
Dreżewski, Rafał 179
Ducatelle, Frederick 121

Edelman, David 228
Eggermont, Jeroen 391
El Hadji, Aboubacar 732
Emmerich, Michael T.M. 391
Esparcia-Alcázar, Anna I. 129, 169

Fan, Kai 189
Farooq, Muddassar 81
Fei, Y.N. 261
Fogelberg, Christopher Graeme 340
Fornari, José 517

Gadomska, Malgorzata 1
Gambardella, Luca Maria 121
García, Gloria 470
García, Jesús 399
Geem, Zong Woo 593
Gervás, Pablo 537
Glette, Kyrre 271
Gómez-Pulido, Juan A. 91, 101
Griffith, Niall J.L. 547

Han, Kijun 153
Hart, David A. 527
Hervás, Raquel 537
Heywood, Malcolm Iain 11
Hochreiter, Ronald 199
Hossain, M. Alamgir 688

Iba, Hitoshi 577

Jaśkowski, Wojciech 281
Ji, T.Y. 367

Kagawa, Tsuneo 439, 459
Kang, Chul-Hee 137
Karkkainen, Tommi 320
Kayacik, H. Gunes 11
Kilic, Sezgin 742
Kim, Hyunsook 153
Kim, Minkyu 21
Kim, Wonsik 21

Kotilainen, Niko 61
Krawiec, Krzysztof 281
Kuan, Ho Chin 557

Labadi, Nacima 722
Laredo, Juan Luís Jiménez 129, 712
Law, Edwin Hui Hean 557
Lenac, Kristijan 375
Lévy Véhel, Jacques 383
Li, M.S. 32
Li, Rui 391
Lim, Hyung-Taig 137
Lipinski, Piotr 208
Lu, Z. 145, 261, 367
Luque, Cristobal 470
Lutton, Evelyne 383
Luyet, Luc 42

Machwe, Azahar T. 449
Maia, Adolfo Jr. 517
Maisto, Domenico 251
Majava, Kirsi 320
Manzolli, Jônatas 477, 517
McDermott, James 547
Médard, Muriel 21
Mendivil, Franklin 383
Merelo, Juan Julián 129, 712
Millan, Cristian 712
Molina, Guillermo 101
Molina, José Manuel 399
Mora, Antonio Miguel 129, 712
Mordonini, Monica 241
Mumolo, Enzo 375

Neri, Ferrante 61, 320
Nishino, Hiroaki 439, 459
Nolich, Massimiliano 375
Nordahl, Mats G. 577

O'Neill, Michael 189, 547
O'Reilly, Una-May 21
O'Sullivan, Conall 189
Olague, Gustavo 291, 407, 423

Pacut, Andrzej 1
Paechter, Ben 129
Pak, Jinsuk 153
Pannese, Lucia 233
Pantano, Pietro 585
Parmee, Ian C. 449
Patricio, Miguel Ángel 399
Pérez, Cynthia Beatriz 407
Pérez, Óscar 399
Phon-Amnuaisuk, Somnuk 557
Piccardi, Massimo 399
Pintea, Camelia M. 702
Poli, Riccardo 301
Pop, Petrica C. 702
Pop Sitar, Corina 702
Priem-Mendes, Silvio 101
Prins, Christian 722
Puiggros, Montserrat 311

Radecker, Matthias 431
Ramirez, Rafael 311
Rand, William 657
Rebelo, Pedro 508
Reghioui, Mohamed 722
Reiber, Johan H.C. 391
Rio, Miguel 71
Riolo, Rick 657
Rizzuti, Costantino 585
Robinson, Jason 537
Rocha, Miguel 71
Rodet, Xavier 488
Romero, Eva 291
Rossi, Tuomo 320
Roth, Martin 121

Sáez, Yago 470
Saint-James, Emmanuel 488
Saleem, Muhammad 81
Sánchez-Pérez, Juan M. 91
Sanjuán, Oscar 470
Santalmasi, Mauro 233
Santamaría, Jose 415
Sartori, Jonathan 241
Saunders, J.R. 32
Scafuri, Umberto 251
Senel, Kerem 218
Seok, Seung-Joon 137
Sepulveda, Francisco 301
Sharman, Ken 169
Shintemirov, Almas 145
Simões, Anabela 617
Siwik, Leszek 179
Son, Jeongho 153
Sousa, Pedro 71
Stathopoulos, Antony 668, 678
Sueyoshi, Takuya 459

Talay, Ahmet Cagatay 161
Tamotsu, Yukihide 439
Tang, W.H. 32, 145, 261
Tang, W.J. 32
Tarantino, Ernesto 251
Tardieu, Damien 488
Tettamanzi, Andrea G.B. 233
Tirronen, Ville 320
Torrecillas, Juan 712
Torresen, Jim 271
Toscano-Pulido, Gregorio 330
Trist, Karen 498
Troc, Maciej 601
Trujillo, Leonardo 291, 423
Tsekeris, Theodore 668, 678

Uludag, Gonul 218
Unold, Olgierd 601
Uozumi, Yuta 609
Utsumiya, Kouichi 439, 459
Uyar, A. Şima 218, 647

Vapa, Mikko 61
Varone, Sacha 42
Vega-Pérez, David 91, 101
Vega-Rodríguez, Miguel A. 91, 101
Vite-Silva, Israel 330, 359
Von Zuben, Fernando 477

Wang, Dingwei 637
Wang, Hongfeng 637
Wieloch, Bartosz 281
Winczura, Katarzyna 208
Wojcik, Joanna 208
Wu, Q.H. 32, 145, 261, 367

Yang, Jun 111
Yang, Shengxiang 627, 637
Yasunaga, Moritoshi 271
Yee-King, Matthew John 567

Zhang, Jie 111
Zhang, Mengjie 340
Zinchenko, Lyudmila 431
Zincir-Heywood, A. Nur 11
Zufferey, Nicolas 42