

*Commenced Publication in 1973*

Founding and Former Series Editors:

Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

## Editorial Board

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

*New York University, NY, 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*



Qing Li Guoren Wang Ling Feng (Eds.)

# Advances in Web-Age Information Management

5th International Conference, WAIM 2004  
Dalian, China, July 15-17, 2004  
Proceedings



Springer

## Volume Editors

Qing Li

City University of Hong Kong

Dept. of Computer Engineering and Information Technology

83 Tat Chee Avenue, Kowloon, Hong Kong SAR, China

E-mail: itqli@cityu.edu.hk

Guoren Wang

Northeastern University, College of Information Science and Engineering

Shengyang 110004, China

E-mail: wanggr@mail.neu.edu.cn

Ling Feng

University of Twente

Faculty of EEMCS, Department of Computer Science, Database Group

7500 AE Enschede, The Netherlands

E-mail: ling@cs.utwente.nl

Library of Congress Control Number: 2004094690

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

ISSN 0302-9743

ISBN 3-540-22418-1 Springer-Verlag 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-Verlag. Violations are liable to prosecution under the German Copyright Law.

Springer-Verlag is a part of Springer Science+Business Media

[springeronline.com](http://springeronline.com)

© Springer-Verlag Berlin Heidelberg 2004

Printed in Germany

Typesetting: Camera-ready by author, data conversion by Olgun Computergrafik

Printed on acid-free paper SPIN: 11018452 06/3142 5 4 3 2 1 0



# Preface

WAIM 2004, Dalian (China), 15–17 July, was hosted by Northeastern University, China and Dalian University of Technology, China in cooperation with the Database Society of CCF. It was the International Conference on Web-Age Information Management. WAIM 2004 brought together participants from universities, industry, and government to present, discuss, and address both current issues and novel approaches in the practice, deployment, theory, and methodology of information management in the context of the World Wide Web. The conference provided an international forum for the sharing of original research results and practical development experiences among researchers and application developers from the many Web-related areas. The previous WAIM conferences were held in Shanghai (WAIM 2000), Xi'an (WAIM 2001), Beijing (WAIM 2002), and Chengdu, China (WAIM 2003). These conferences turned out to be very successful and each attracted more than 100 participants from all over the world.

The WAIM 2004 conference included technical sessions organized around important subtopics such as data stream processing, time series data processing, mobile computing, cache management, security, XML, Web services, Web search engines, query evaluation, and data mining, etc. Following careful review of the 291 submissions by members of the international program committee, 57 regular papers and 23 short/industrial papers were accepted for presentation at the conference and for publication in this volume. The submissions came from a total of 22 countries and regions, and the accepted ones were from 13 countries and regions, including Australia, Canada, Chile, China, Hong Kong SAR, Korea, Norway, Singapore, Spain, Taiwan, UAE, UK, and USA. The conference also consisted of a poster and demonstration track, the publication of which was done in CD-ROM form.

A conference such as this can only succeed as a team effort. We would like to thank program committee members and reviewers for their efforts, and the WAIM steering committee members for their invaluable input and advice. We thank the conference chairs Prof. Wang Shan and Tok Wang Ling whose involvement and support added greatly to the quality of the conference. Our sincere gratitude goes to all of the authors who submitted papers. We are grateful to our sponsors for their generous support. Special thanks go to Prof. Yu Ge who served as both Local Arrangements Co-chair and Finance and Registration Chair, Prof. Katsumi Tanaka who selected and organized high-quality tutorials, Profs. Zhiyong Peng, Arne Solvberg, and Mengchi Liu for their effective services as Publicity Co-chairs, and Drs. Mukesh Mahania and Ji-Rong Wen who organized the industrial track and the poster and demonstration track, respectively.

On behalf of the Organizing and Program Committees of WAIM 2004, we trust the attendees found the conference a fruitful experience and that they had an enjoyable stay in the beautiful city of Dalian, China.



# Organization

WAIM 2004 was organized by Northeastern University, China and Dalian University of Technology, China in cooperation with the China Computer Federation, Database Society and ACM SIGMOD. It was sponsored by Fujitsu Ltd., Japan, Oracle Corporation, China, and SicNoah Software Technology Co., Ltd., and the National Science Foundation, China

## Executive Committee

Conference Co-chairs	Wang Shan (Renmin University, China) Tok Wang Ling (NUS, Singapore)
Program Co-chairs	Qing Li (City University, HK SAR, China) Guoren Wang (Northeastern University, China)
Local Organization Co-chairs	Yu Ge (Northeastern University, China) Xiukun Wang (Dalian University Tech, China)
Publication Chair	Feng Ling (University of Twente, Netherlands)
Finance and Registration Chair	Yu Ge (Northeastern University, China)
Publicity Co-chairs	Peng Zhiyong (Asia-Pacific) Arne Solvberg (Europe) Mengchi Liu (America)
Tutorial Chair	Katsumi Tanaka (Kyoto University, Japan)
Panel Chair	Ee-Peng Lim (NTU, Singapore)
Industrial Track Chair	Mukesh K. Mohania (IBM India)
Exhibition and Demo Chair	Ji-Rong Wen (Microsoft Research Asia)
Steering Committee Liaison	Hongjun Lu (HKUST, HK SAR, China)

## Program Committee

Stephane Bressan	National University of Singapore, Singapore
Ying Cai	Iowa State University, USA
Keith Chan	Hong Kong Polytechnic University, Hong Kong SAR
Zheng Chen	Microsoft Research Asia, Beijing, China
David Cheung	University of Hong Kong, Hong Kong SAR
Dickson K.W. Chiu	Dickson Computer Systems, Hong Kong SAR
Gill Dobbie	University of Auckland, New Zealand
Xiaoyong Du	Renmin University of China, China
David Dagan Feng	University of Sydney, Australia
Jianhua Feng	Tsinghua University, China
Vivekanand Gopalkrishnan	NTU, Singapore
Hong Gao	Harbin Institute of Technology, China
Jingfeng Guo	Yanshan University, China
Yanbo Han	Institute of Computing Technology, Chinese Academy of Sciences, China
Yanxiang He	Wuhan University, China
Kien A. Hua	University of Central Florida, USA
Richard Hull	Bell Labs Research, Lucent, USA
Patrick Hung	CSIRO, Australia
Xiao Ji	Shanghai Baosight Ltd. Company, China
Yan Jia	National University of Defence Technology, China
Qingshan Jiang	Xiamen University, China
Kamal Karlapalem	International Institute of Information Technology, India
Irwin King	Chinese University of Hong Kong, Hong Kong SAR
Jiajin Le	Donghua University, China
Dik Lun Lee	HKUST, Hong Kong SAR
YoonJoon Lee	KAIST, Korea
Hong Va Leong	Hong Kong Polytechnic University, Hong Kong SAR
Jianzhong Li	Harbin Institute of Technology, China
Tianzhu Li	Hebei University, China
Zhanhuai Li	Northwestern Polytechnic University, China
Yaping Lin	Huana University, China
Huan Liu	Arizona State University, USA
Mengchi Liu	Carleton University, Canada
Wenyin Liu	City Univ. of Hong Kong, Hong Kong SAR
Weiyi Liu	YunNan University, China
Frederick H. Lochovsky	HKUST, Hong Kong SAR
Akifumi Makinouchi	Kyushu University, Japan
Dennis McLeod	University of Southern California, USA

Xiaofeng Meng	Renmin University of China, China
Maria Orlowska	University of Queensland, Australia
Zhiyong Peng	Wuhan University, China
Prasan Roy	IBM India Research Lab, India
Junyi Shen	Xi'an Jiaotong University, China
Baile Shi	Fudan University, China
John R. Smith	IBM T.J. Watson Research Center, USA
Jianwen Su	University of California, Santa Barbara, USA
Changjie Tang	Sichuan University, China
Shiwei Tang	Peking University, China
Yufei Tao	City Univ. of Hong Kong, Hong Kong SAR
Wallapak Tavanapong	Iowa State University, USA
Duc A. Tran	University of Dayton, USA
Christelle Vangenot	Swiss Federal Institute of Technology, Switzerland
Daling Wang	Northeastern University, China
Min Wang	IBM T.J. Watson Research Center, USA
Tengjiao Wang	Peking University, China
Xiukun Wang	Dalian University of Technology, China
X. Sean Wang	University of Vermont, USA
Raymond Wong	University of New South Wales, Australia
Chunxiao Xing	Tsinghua University, China
Christopher C. Yang	Chinese University of Hong Kong, Hong Kong SAR
Dongqing Yang	Peking University, China
Jian Yang	University of Tilburg, The Netherlands
Masatoshi Yoshikawa	Nagoya University, Japan
Ge Yu	Northeastern University, China
Jeffrey X. Yu	Chinese University of Hong Kong, Hong Kong SAR
Lihua Yue	University of Science and Technology, China
Yanchun Zhang	Victoria University, Australia
Aoying Zhou	Fudan University, China
Lizhu Zhou	Tsinghua University, China
Shuigeng Zhou	Fudan University, China
Xiaofang Zhou	University of Queensland, Australia
Yueting Zhuang	Zhejiang University, China
Hai Zhuge	Institute of Computing Technology, Chinese Academy of Sciences, China

## External Referees

Abhishek Verma	Jun He	Srihari Venkatesan
Amit Mandvikar	Ken C.K. Lee	Sufang Hou
Bin Zhou	Krishna Prasad	Tok Wee Hyong
Cagdas Gerede	Chitrapura	Wai-Shing Ho
Cheong Fung	Kun Yue	Xiaoling Wang
Chung-yi Chang	Lance R. Parsons	WeiHong Han
Daoshun Wang	Laurent Mignet	Weining Qian
Deng Ke	Lei Yi	Weixia Wei
Edward Ho	Lei Yu	Xiang Fu
Ehtesham Haque	Leila Kaghazian	Xuan Tian
Fang Ying	Liang Zhu	Xue Li
Gabriel Pui	Luo Yi	Xueqing Gong
Hoda M.O. Mokhtar	Manish Bhide	Xuhui Li
Indra Budi	Mohamed Rafi	Yang Cao
J. Ravindranath	Nadine Culot	Yang Xiaochun
James Liu	Omar Boucelma	Yanyan Guo
Jhansi Rani V	Panagiotis Kalnis	Yifeng Gao
Jian Zhai	Qi Guo	Ying Cai
Jianpo Ou	Qingqing Yuan	Yu Li
Jigar Mody	Sai Sun	Yu Wang
Jing Sun	Sang-Soo Sung	Yuan Yuan
Jingyu Hou	V.R. Satyanarayana	Zeng Cheng
Jiying Wang	Seoungwook Youn	Zhimao Guo
Jong-eun Jun	Sham Prasher	Zhiqiang Zhang
Joshua Pun	Shijun Li	M. Zimmer
Juanzi Li	V. Soujanya	

# Table of Contents

## Invited Talks

A Framework for a Trusted Environment for Virtual Collaboration . . . . .	1
<i>Tharam S. Dillon, Elizabeth Chang, and Farookh Hussain</i>	
Future Interconnection Environment – Dream, Principle, Challenge and Practice . . . . .	13
<i>Hai Zhuge</i>	
Recent Advances in Histogram Construction Algorithms . . . . .	23
<i>Kyuseok Shim</i>	

## Data Stream Processing

Dynamic Adjustment of Sliding Windows over Data Streams . . . . .	24
<i>Dongdong Zhang, Jianzhong Li, Zhaogong Zhang, Weiping Wang, and Longjiang Guo</i>	
Space Efficient Quantile Summary for Constrained Sliding Windows on a Data Stream . . . . .	34
<i>Jian Xu, Xuemin Lin, and Xiaofang Zhou</i>	

## Time Series Data Processing

The Dimension-Reduced Multi-scale Histogram: A New Method for Fast and Efficient Retrieval of Similar Time Sequences and Subsequence . . . . .	45
<i>Jian-Wei Liu, Shou-Jian Yu, and Jia-Jin Le</i>	
Time Series Prediction Based on Gene Expression Programming . . . . .	55
<i>Jie Zuo, Chang-jie Tang, Chuan Li, Chang-an Yuan, and An-long Chen</i>	
A Grid-Based Index Method for Time Warping Distance . . . . .	65
<i>Jiuyan An, Yi-Ping Phoebe Chen, and Eamonn Keogh</i>	

## Security

Fail-Stop Authentication Protocol for Digital Rights Management in Adaptive Information Retrieval System . . . . .	76
<i>Zhaofeng Ma and Boqin Feng</i>	
Effective Web-Related Resource Security Using Distributed Role Hierarchy . . . . .	87
<i>Gunhee Lee, Wonil Kim, Dong-kyoo Kim, and Hongjin Yeh</i>	

## Mobile Computing

A Low-Cost Checkpointing Scheme for Mobile Computing Systems . . . . .	97
<i>Guohui Li, Hongya Wang, and Jixiong Chen</i>	
Semantic and Adaptive Middleware for Data Management in Smart Vehicle Space . . . . .	107
<i>Qing Wu, Zhaohui Wu, Bin Wu, and Zhou Jiang</i>	
Dynamic Resource Reservation Using the Differentiated Handoff Estima- tion Model for Mobile Networks . . . . .	117
<i>Si-Yong Park and Ki-Dong Chung</i>	
Adaptive Generic Communications for Integrated Mobile and Internet Web-Services . . . . .	127
<i>Man-Ching Yuen, Leung Cheng, Pui-On Au, and Weijia Jia</i>	
Spatio-temporal Database with Multi-granularities . . . . .	137
<i>Sheng-sheng Wang and Da-you Liu</i>	

## Cache Management

A Simple but Effective Dynamic Materialized View Caching . . . . .	147
<i>Chi-Hon Choi, Jeffrey Xu Yu, and Hongjun Lu</i>	
A Cache Replacement Algorithm in Hierarchical Storage of Continuous Media Object . . . . .	157
<i>Yaoqiang Xu, Chunxiao Xing, and Lizhu Zhou</i>	
CCAN: Cache-Based CAN Using the Small World Model . . . . .	167
<i>Futai Zou, Yin Li, Liang Zhang, and Fanyuan Ma</i>	
Performance Evaluations of Replacement Algorithms in Hierarchical Web Caching . . . . .	176
<i>Haohuan Fu, Pui-On Au, and Weijia Jia</i>	
Component-Based WebGIS and Its Spatial Cache Framework . . . . .	186
<i>Yingwei Luo, Xiaolin Wang, and Zhuoqun Xu</i>	

## Query Evaluation

Learning-Based Top-N Selection Query Evaluation over Relational Databases . . . . .	197
<i>Liang Zhu and Weiyi Meng</i>	
DHT Based Searching Improved by Sliding Window . . . . .	208
<i>Shen Huang, Gui-Rong Xue, Xing Zhu, Yan-Feng Ge, and Yong Yu</i>	



A Scalable and I/O Optimal Skyline Processing Algorithm . . . . .	218
<i>Yi Luo, Hai-Xin Lu, and Xuemin Lin</i>	
Linearization Approach for Efficient KNN Search of High-Dimensional Data . . . . .	229
<i>Zaher Al Aghbari and Akifumi Makinouchi</i>	
SQL/MDR: Query Language for Consistent Access of Metadata Registries Based on SQL3 . . . . .	239
<i>Dongwon Jeong, Young-Gab Kim, and Doo-Kwon Baik</i>	
Solving Queries over Semantically Integrated Biological Data Sources . . . . .	249
<i>José Francisco Aldana-Montes, Ismael Navas-Delgado, and María del Mar Roldán-García</i>	
Fast Approximate Search in Text Databases . . . . .	259
<i>Fei Shi</i>	

## Web Search Engine

Apply Feedback Control Theory to Design Soft Real-Time Search Engine System . . . . .	268
<i>Huayong Wang and Yiqi Dai</i>	
User Interest Detection on Web Pages for Building Personalized Information Agent . . . . .	280
<i>Yin Liu, Wenyin Liu, and Changjun Jiang</i>	
Improved Link-Based Algorithms for Ranking Web Pages . . . . .	291
<i>Ziyang Wang</i>	
Effectiveness Evaluation and Comparison of Web Search Engines and Meta-search Engines . . . . .	303
<i>Shengli Wu and Jieyu Li</i>	
Semantic Search in the WWW Supported by a Cognitive Model . . . . .	315
<i>Katia Wechsler, Jorge Baier, Miguel Nussbaum, and Ricardo Baeza-Yates</i>	
Semantic Board: A Bulletin Board System Supporting Dynamic Semantic Information . . . . .	325
<i>Eui-Hyun Jung</i>	
Efficient Evaluation of Sparse Data Cubes . . . . .	336
<i>Lixin Fu</i>	

## XML

Redundancy Free Mappings from Relations to XML .....	346
<i>Millist W. Vincent, Jixue Liu, and Chengfei Liu</i>	
Tree Multivalued Dependencies for XML Datasets .....	357
<i>Lawrence V. Saxton and Xiqun Tang</i>	
TROS: A Novel Model and Its Implementation for XML Data Management .....	368
<i>Wenwen Qi and Aoying Zhou</i>	
Optimized Query Translation Strategy for XML Stored in Relational Database .....	378
<i>Hongzhi Wang, Jianzhong Li, and Zhenying He</i>	
QReduction: Synopsizing XPath Query Set Efficiently under Resource Constraint .....	389
<i>Jun Gao, Xiuli Ma, Dongqing Yang, Tengjiao Wang, and Shiwei Tang</i>	
Extracting Key Value and Checking Structural Constraints for Validating XML Key Constraints .....	399
<i>Yunfeng Liu, Dongqing Yang, Shiwei Tang, Tengjiao Wang, and Jun Gao</i>	
Estimating the Selectivity of XML Path Expression with Predicates by Histograms .....	409
<i>Yu Wang, Hairun Wang, Xiaofeng Meng, and Shan Wang</i>	
Wrapping Web Pages into XML Documents .....	419
<i>Tao Fu</i>	

## Web Services

Rapid Prototyping for Web Applications .....	429
<i>Chee Chern Lim, Man Hing Yu, and Jesse J. Jin</i>	
Capacity Planning for Composite Web Services Using Queueing Network-Based Models .....	439
<i>Dunlu Peng, Yang Yuan, Kun Yue, Xiaoling Wang, and Aoying Zhou</i>	
A Web-Based Coordination Infrastructure for Grid Collective Services ...	449
<i>Javier Jaén, Jose H. Canós, and Elena Navarro</i>	
Symbolic Agent Negotiation for Semantic Web Service Exploitation .....	458
<i>Peep Kungas, Jinghai Rao, and Mihhail Matskin</i>	

Object-Oriented Realization of Workflow Views for Web Services – An Object Deputy Model Based Approach .....	468
<i>Zhe Shan, Zhiyi Long, Yi Luo, and Zhiyong Peng</i>	
Modeling QoS for Semantic Equivalent Web Services .....	478
<i>Derong Shen, Ge Yu, Tiezheng Nie, Rui Li, and Xiaochun Yang</i>	

## Classification

Improved Email Classification through Enriched Feature Space .....	489
<i>Yunming Ye, Fanyuan Ma, Hongqiang Rong, and Joshua Zhexue Huang</i>	
PLD: A Distillation Algorithm for Misclassified Documents .....	499
<i>Ding-Yi Chen and Xue Li</i>	
Sequential Classifiers Combination for Text Categorization: An Experimental Study .....	509
<i>Zheng Zhang, Shuigeng Zhou, and Aoying Zhou</i>	
Combining Active Learning and Boosting for Naïve Bayes Text Classifiers .....	519
<i>Han-joon Kim and Je-uk Kim</i>	

## Data Mining

Using Interval Association Rules to Identify Dubious Data Values .....	528
<i>Ren Lu, Mong Li Lee, and Wynne Hsu</i>	
Mining Web Sequential Patterns Incrementally with Revised PLWAP Tree .....	539
<i>Christie I. Ezeife and Min Chen</i>	
Mining Frequent Items in Spatio-temporal Databases .....	549
<i>Cheqing Jin, Fang Xiong, Joshua Zhexue Huang, Jeffrey Xu Yu, and Aoying Zhou</i>	
Discovery of Frequent XML Query Patterns with DTD Cardinality Constraints .....	559
<i>Yunfeng Liu, Dongqing Yang, Shiwei Tang, Tengjiao Wang, and Jun Gao</i>	
Mining Inheritance Rules from Genealogical Data .....	569
<i>Yen-Liang Chen and Jing-Tin Lu</i>	
Mining DAG Patterns from DAG Databases .....	579
<i>Yen-Liang Chen, Hung-Pin Kao, and Ming-Tat Ko</i>	

Mining Class Outliers: Concepts, Algorithms and Applications . . . . .	589
<i>Zengyou He, Joshua Zhexue Huang, Xiaofei Xu, and Shengchun Deng</i>	
CD-Trees: An Efficient Index Structure for Outlier Detection . . . . .	600
<i>Huanliang Sun, Yubin Bao, Faxin Zhao, Ge Yu, and Daling Wang</i>	

## Industrial/Short Papers

Mobile Data Management with Sound . . . . .	610
<i>Ook Lee</i>	
A Mixture Model of Classical Fuzzy Classifiers . . . . .	616
<i>Yongchuan Tang and Shouqian Sun</i>	
An Empirical Study of Building Compact Ensembles . . . . .	622
<i>Huan Liu, Amit Mandvikar, and Jigar Mody</i>	
Merging Individually Learned Optimal Results to Accelerate Coordination . . . . .	628
<i>Huaxiang Zhang and Shangteng Huang</i>	
The Compression of Massive Offline Relations . . . . .	634
<i>Jizhou Luo, Jianzhong Li, Hongzhi Wang, Yanqiu Zhang, and Kai Zhao</i>	
Finding Plagiarism Based on Common Semantic Sequence Model . . . . .	640
<i>Jun-Peng Bao, Jun-Yi Shen, Xiao-Dong Liu, Hai-Yan Liu, and Xiao-Di Zhang</i>	
Web Information Extraction Based on Similar Patterns . . . . .	646
<i>Na Ye, Xuejun Wu, Jingbo Zhu, Wenliang Chen, and Tianshun Yao</i>	
TS-Cache: A Novel Caching Strategy to Manage Data in MANET Database . . . . .	652
<i>Shengfei Shi, Jianzhong Li, Chaokun Wang, and Jinbao Li</i>	
Extended Chain-Conflicting Serializability for the Correct Schedule of Transactions in Hierarchical Multidatabase . . . . .	658
<i>Guoning Chen and Taoshen Li</i>	
An Efficient Hierarchical Failure Recovery Algorithm Ensuring Semantic Atomicity for Workflow Applications . . . . .	664
<i>Yi Ren, Quanyuan Wu, Yan Jia, and Jianbo Guan</i>	
A Macrocommittees Method of Combining Multistrategy Classifiers for Heterogeneous Ontology Matching . . . . .	672
<i>Leyun Pan, Hui Song, and Fanyuan Ma</i>	

Hypertext Classification Algorithm Based on Co-weighting Multi-information .....	678
<i>Ya Peng, Ya-ping Lin, and Zhi-ping Chen</i>	
Verifying a MAC-Based Service Bundle Authentication Mechanism for the OSGi Service Platform .....	684
<i>Young-Gab Kim, Dongwon Jeong, and Doo-Kwon Baik</i>	
Optimal Data Dispatching Methods in Near-Line Tertiary Storage System.....	690
<i>Baoliang Liu, Jianzhong Li, and Yanqiu Zhang</i>	
A Real-Time Information Gathering Agent Based on Ontology .....	696
<i>Jianqing Li, Shengping Liu, ZuoQuan Lin, and Cen Wu</i>	
Dynamical Schedule Algorithms Based on Markov Model in Tertiary Storage .....	702
<i>Yanqiu Zhang, Jianzhong Li, Zhaogong Zhang, and Baoliang Liu</i>	
Image Retrieval Using Structured Logical Shape Feature .....	708
<i>Nanhyo Bang and Kyhyun Um</i>	
Automatic HTML to XML Conversion.....	714
<i>Shijun Li, Mengchi Liu, Tok Wang Ling, and Zhiyong Peng</i>	
Quality Improvement Modeling and Practice in Baosteel Based on KIV-KOV Analysis .....	720
<i>Xinyu Liu, Haidong Tang, Zhiping Fan, and Bing Deng</i>	
A Frequent Pattern Discovery Method for Outlier Detection .....	726
<i>Zengyou He, Xiaofei Xu, Joshua Zhexue Huang, and Shengchun Deng</i>	
Adaptive Load Balancing and Fault Tolerance in Push-Based Information Delivery Middleware Service .....	733
<i>Zhaofeng Ma and and Boqin Feng</i>	
Performance Optimization of Fractal Dimension Based Feature Selection Algorithm .....	739
<i>Yubin Bao, Ge Yu, Huanliang Sun, and Daling Wang</i>	
Graph-Based Cyclic Multi-join Query Optimization Algorithm .....	745
<i>Hong Yu, Xiu-Kun Wang, and Jian-Ying Zhang</i>	
<b>Author Index .....</b>	<b>751</b>

# A Framework for a Trusted Environment for Virtual Collaboration

Tharam S. Dillon<sup>1</sup>, Elizabeth Chang<sup>2</sup>, and Farookh Hussain<sup>2</sup>

<sup>1</sup> Faculty of Information Technology, University of Technology, Sydney, Australia  
tharam@it.uts.edu.au

<sup>2</sup> School of Information Systems, Curtin University of Technology, Australia  
{ChangE,HussainF}@cbs.curtin.edu.au

**Abstract.** Trusted computing has assumed major significance since the advent of internet based commerce and peer to peer communications. Trust is the belief that a peer has another peer in a given context. This paper addresses both identity trust which establishes the identity of another party and behaviour trust which answers the question ‘Do you believe that the entity will behave as you expect it to?’ It develops complete definitions of context and time dependant trust and reputation and develops trust models, trust relationships of a peer.

## 1 Introduction

Trust has been a focal point in interpersonal relationships in many domains including Business, Sociology, and Psychology etc. Recently there has been an upsurge of interest in Trust in the Computer Science [1, 2, 3, 5, 7, 10-18, 26] research community, and its applications in the Peer-to-Peer (P2P) domain. One important distinction between trust in other fields and in computer science is that the first group often involves physical environments whilst the second involves virtual environments. In physical environments, one can use various physical or facial cues or documents or referral to known such as credit agencies or government to inform the process of trust establishment [3]. In virtual environments, there is an absence of physical cues, the establishment of centralized authorities such as certification authorities is still evolving for referral and may not be always applicable. Four factors have created some urgency in tackling the issues in trust and these are the advent of (i) peer-to-peer communication which could be between anonymous peers, (ii) virtual communities with the need to protect the integrity of the community, (iii) e-commerce which involves business transactions on the internet and (iv) lastly cyber-terrorism which aims at disrupting services. In this paper we propose a conceptual structure and a trust model.

## 2 Types of Trust and Security

Marsh first introduced the notion of trust in Distributed Artificial Intelligence [10]. However, Trust became really important with the widespread use of the Internet,

particularly with the advent of business transactions on it. If two interacting parties do not trust each other, they will be reluctant to carry out business transactions with each other [27]. Hence we need to investigate how trust between them can be established. Security and trust are two distinct concepts and hence should be considered separately. Trust is the belief or faith that a peer has in another peer in a given context [27,30], that gives him/her confidence to carry out a transaction with the trusted peer. In the virtual world, in order to acquire trust in another entity, security establishing mechanisms are used [12, 14]. Security in the virtual world usually refers to the process of enabling sheltered communication between two communicating entities. Lack of trust between communicating peers, results in the prisoner's dilemma, in which parties could resort to unfair practices that may be difficult to detect until the completion of the transaction. We need a mechanism for the two communicating parties to develop trust in each other, since security can enable only secure communication between them. Various methods/infrastructures exist to ensure secure communication which we call *Security Establishing Mechanisms*, and they [6]: 1) Help both the communicating parties authenticate themselves; 2) Help a party to prove to its counterpart that it is authorized to carry out the activity; and 3) To provide a means by which information can be exchanged securely over the internet. Hence these mechanisms can be used by a party to establish the identity of another party. We call this type of trust in the identity of another party as *Identity Trust*. Another type of trust is *Behaviour Trust*, which answers the question; "Do you believe that the entity will behave as you expect it to?" Another type of trust that can exist between two peers is Hybrid Trust [27], which characterizes both identity trust and behaviour trust.

The major problem with the existing methods proposed for establishing trust between two peers is that they either propose a method to only establish identity trust or to only establish behaviour trust between two peers. None of them propose a method of establishing both identity trust and behaviour trust of a peer, and it is not clear that the existing trust models and trust protocols are compatible with each other as each of them have their own assumptions which contradict the assumptions of the others. To the best of our knowledge, there is no work on trust relationships. We present a definition of what a trust relationship is and define the different possible types of trust relationships between peers in P2P Communication. Another problem with many methods for establishing identity trust between two peers is that they assume the existence of a central authority that can authenticate the identity of another entity[5,14], but they do not address issues like the fault-tolerance of the central authority, load balancing at the central authority, etc.. This assumption of a central trusted authority also makes the system vulnerable against inappropriate behaviour by the central trusted authority. Our proposed method is partially decentralized.

Some trust models have been proposed to establish trust between two communicating peers.[2,5,7,11,15,16,17,18,26]. However with the exception of [11] all of the proposed trust models do not consider the context specific and dynamic nature of trust. Additionally [11] only considers the context specific nature of trust, but not the dynamic nature of trust.

Some trust management protocols have been proposed to establish trust between two communicating peers. [5,12,13,14]. [12] proposes the use of an anonymous P2P

trust management protocol. However, it is vulnerable to attacks like Repudiation Attacks, Self-trust Insertion Attacks. They do not propose any solution for counteracting the problem for multiple identities of peers (Clique Attacks). Moreover they propose the use of a single server for authenticating the identity of a peer and, therefore, has the problems referred to above. Additionally they focus on an anonymous protocol but they do not provide a non-anonymous and pseudonymous protocol for establishing trust between two peers. [14] proposes the use of a pseudonymous trust management protocol for establishing identity trust between two communicating peers. This protocol solves the problem of peer authentication when the identities of the peers are pseudonymous. However it is vulnerable to Clique Attacks and Self-Trust Insertion Attacks. They do not need any single server for the proper functioning of their proposed protocol. However they focus on the pseudonymous protocol but they do not provide a non-anonymous and anonymous protocol for establishing trust between two peers.

Most of the P2P communication, involving final financial transactions, is non-anonymous. From the above, we note that there is no non-anonymous P2P trust management infrastructure in the literature. Moreover the anonymous and pseudonymous versions of the trust protocol are vulnerable to several attacks. Additionally Security approaches, like Cryptography or Stenography can only enable secure communication between the involved peer but they CANNOT establish trust or belief between the communicating peers.

We also need for a method to help a trusting peer avoid the possibility of interacting with malicious peers. Most of the existing proposed trust models, do not consider the RISK factor involved in an interaction before it occurs helping a peer make a decision of whether to interact with another peer. In our proposed trust support system, we model risk in an interaction with a trusted peer and based on the Expected Value of the transaction(which takes in to account the risk involved) we propose a trust support system. Apart from risk factor we take into account various other factors that need to be addressed. To trust somebody is to take a risk and make themselves vulnerable to their actions [9].

### **3 Definition of the Conceptual Framework for Trust, Reputation in Peer-to-Peer Communications and Trust Models**

This issue will be addressed in four different steps:

#### **3.1 Definitions of Trust**

Several definitions for Trust and Reputation have appeared in the literature [28], [10].

These existing definitions of trust are vague and incomplete and do not capture all the aspects of trust in P2P communications. [18] It is also important to understand the characteristics of peer-to-peer communication when developing this definition of trust. These characteristics include: anonymity / non-anonymity / pseudonymity of



peers, the decentralised nature of communication and the heterogeneous nature of individual peers.

For further discussion we make use of the terms *trusted peer* and *trusting peer*. Our definition of trust for P2P communications is as follows.

The belief that the trusting peer has in the willingness and capability of the trusted peer, to behave in a given context, and at a given point of time, as expected by trusting peer.[27,30] The notions of ‘belief’, ‘context’, ‘point of time’, ‘willingness and capability’ and ‘as expected by another peer’ are very important in the definition of trust. We define a trusting peer as *a human entity who controls resources and who reposes his<sup>1</sup> faith in some other human entity*. In other words, an entity who has faith or belief in another entity takes up the role of the trusting peer and a trusted peer as *a human entity who controls resources and in whom faith is reposed by some other entity*. Consider for example, Alice trusts Bob with his credit card details. The context in which Alice reposes faith in Bob is “*handing over credit card details*”. This is the scenario or situation in which Alice trusts Bob. In other scenarios or situations Alice may or may not trust Bob. The term *context* is significant while defining and considering trust.

When the trusting peer reposes trust in the trusted peer in a given context, he has a certain idea or notion of how the trusted peer will conduct itself in that context in which trust is reposed. We term this anticipated behaviour of the trusted peer as the “Expected Behaviour” [27].

The term “*a given time period*”, in the definition of trust captures the dynamic nature of trust. Our work uses this definition of trust. Classify and identifies the different types of Trust in Peer-to-Peer Communication [27]. A classification of different types of trust in P2P communication would give us an idea of what types of trust should our proposed infrastructure be able to establish between two communicating peers.

### 3.2 Definition of Reputation

We present a formal definition of Reputation. Similar to trust, reputation and related concepts to reputation have been either loosely or incompletely defined in the existing literature on reputation in Computer Science. We give a definition of what we mean by reputation. Again it is necessary to include the notions of context specific, dynamic with time, and the element of historical behaviours in these definitions of reputation. Reputation plays a pivotal role in the process of establishing trust between communicating peers. If the two peers have previously interacted with each other, then they know the trustworthiness of each other. Based on this previous knowledge of each others’ trustworthiness the interacting peers can decide whether to interact with each other in future or not.

However usually the two interacting peers may not have interacted with each other previously and hence they do not know the trustworthiness of each other. Reputation mechanisms help in filling this gap by

- Providing the trusted peer an idea of the trustworthiness of the trusting peer
- Providing the trusting peer an idea of the trustworthiness of the trusted peer

In this section we present a formal definition of reputation for peer-to-peer communication. Additionally we also discuss some other definitions of reputation in literature and discuss why these definitions are not appropriate for peer-to-peer communication.

Before we present a definition of reputation we will define some terms that we make use of through out this paper.

We define Interacting peers, *“as peers who are communicating or having a transaction with each in order to attain certain objectives”*

We define Reputation Querying Peer, *“as the peer who is querying about the reputation of another peer”*

We define Reputation Queried Peer, *“as peer whose reputation is being queried by the Reputation Querying Peer”*

We define Witness Peer (or) Reputation Responding Peer, *“as a peer who knows about the trustworthiness of the reputation queried peer based on its direct interaction with the reputation queried peer”*

We define a Reputation Query *“as a query about the reputation of the reputation queried peer posed by the reputation querying peer”*. We note the following points about Reputation Query

- It is associated with exactly one reputation querying peer
- It may be associated with more than one reputation queried peer
- One or more than one witness peer may reply to the reputation query.

Not every peer will know about the trustworthiness of the reputation queried peer. Usually if a peer does not know about the trustworthiness of the reputation queried it passes the reputation query to peers whom it feels may know the trustworthiness of the reputation queried peer. We define Intermediate Peers *“as peers who pass the reputation query to other peers”*.

For explanation purposes we take the example of five peers, namely Peer A, Peer B, Peer C, Peer D and Peer E and the trust relationships between them through this paper. Let us assume that Peer A had previously interacted with Peer B and Peer E. Furthermore let us assume that Peer B has previously interacted with Peer C. We define reputation of a peer as *“the perceived trustworthiness of the reputation queried peer as advised by the witness peers, in a given context and at a given point of time”*. The phrase *“witness peers”* signify which peers are eligible to vouch for the reputation of the reputation queried peer. We feel that only the peers who have had direct interaction with the reputation queried peer can vouch for the reputation of the reputation queried peer.

The relationship between trustworthiness and reputation can be summarized as follows

*Trustworthiness assigned by the trusting peer to the trusted peer, becomes reputation of the trusted peer, when the trusting peer conveys this to other peers.*

Using reputation of a peer is a chief method of establishing trust with the peer to classify and identify the different types of Reputation in Peer-to-Peer Communication

in our trust model. The types of reputation that we identify in P2P will enable us to identify what types of reputation need to be taken into consideration when making a decision of whether to trust another peer.

### 3.3 An Ontology Based Approach to Allow Shared Conceptualization of Contexts for Trust

We model trust in a given scenario independent of trust in other scenarios. To the best of our knowledge all the existing trust models do not model trust in a given situation in isolation from the trust in the other situations, i.e. they are not context-dependent. We will make use of ontology for having a shared understanding of the different contexts. In a distributed and decentralized environment like P2P communication, it is possible that the same context is referred to in different ways. The only way we can have a shared understanding of the different contexts is with an ontology. In order to enable the comparison of different services or different contexts, we propose the use of an ontology. The Ontology provides semantic meaning to a context and helps to have a shared understanding of a given context, between different peers, which in a distributed decentralized environment is difficult to achieve.

In a distributed environment like Peer-to-Peer Communication, it is possible that the different peers may use a different terminology to refer to the same context. In such a distributed and diverse environment it is not possible to enforce a single terminology for a given context.

As an example, consider a transaction where a logistics company A wants to use the warehouse of B located at Detroit, after which it assigned a particular trustworthiness value and the context as 'Storing the goods'. If another peer asks Peer A about the reputation of Peer B in the context of 'Renting warehouse space', Peer A may not be sure if the reputation querying peer is referring to the same context as 'Storing the goods'. We propose the use of an ontology for mapping the various terminology used to refer to a given context as it is not possible otherwise to reach a consensus for referring to a given context in a distributed environment like peer-to-peer communication. It can also help put caveats and constraints on the context. The ontology provides semantics to a context and a shared understanding of the context.

## 4 A Generic Trust Management Protocol for Peer-to-Peer(P2P) Communication

The second research issue of this project is to create a framework for managing the identity of a peer (Identity Trust). Identity Trust is defined as the *trust that the trusting peer has in the trusted peer that indicates the extent to which the trusting peer believes that the identity claimed by the trusted peer, belongs to it. Our proposed trust management protocol will enable a peer to ensure that identity of another peer. Our proposed framework for establishing Identity Trust will be generic to be able to ap-*

plicable to Anonymous Peer-to-Peer Communication, Pseudonymous Peer-to-Peer Communication and Non-Anonymous Peer-to-Peer Communication.

The most crucial phase is to ensure that the Trust Management Protocol is resilient to attacks like Self-Trust Insertion Attacks, Sybil Attack, and Clique Attacks apart from nullifying the possibility of Repudiation Attacks, Ensuring Integrity of Message, Impersonation Attacks and Replay attacks.

This protocol should be partially decentralized/completely decentralized. Additionally it should be portable to most/all P2P applications irrespective of the domain of the application.

When making a trusting decision of whether to trust another peer; various factors need to be considered. In this phase we will identify all the factors that a peer needs to consider when making a trusting decision. Based on the factors identified, this we intend to develop a trust support system with the trust management protocol and the trust model as its base, which will enable a peer to make a decision of whether to trust another peer or not. Risk involved in interacting with a given trusted peer (say A) will be assessed. The outcome of the previous interaction with the trusted peer (say A) and the recommendations received from other peers whom the trusting peer trusts to give recommendations, will be utilised.

## **5 A Generic Framework for Assigning a Trustworthiness Value to a Peer after Interaction**

We will create a generic framework with the help of which the trusting peer can assign a trustworthiness value to the trusted peer. This is an area which has been untouched in Literature. We will develop mathematical representations in terms of rough set theory to assign a trustworthiness value to a peer.

This proposed approach will be generic enough to be able to be applicable to any interaction in Multi-Agent Systems, Peer-to-Peer Communication, Client-Server Communication, etc. When calculating this value of trustworthiness, it will be important to take account of the various factors that influence the decision to trust.

We intend to develop a methodology for assigning trustworthiness value to a peer. We will develop an algorithm which models trust and helps a trusting peer assign a certain degree of trustworthiness to another peer, based on the mathematical model of trustworthiness that we develop. Additionally, once the algorithm is developed, we will use it in our proposed trust model and the prototype implementation in order to determine proper functioning and to provide proof of concept.

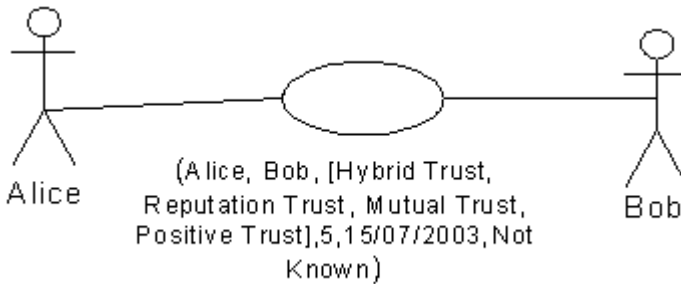
## **6 Peer to Peer Trust Modelling**

Modeling is a method of representing reality. A model is created to produce a simplified version of a system. Additionally, a model of a system is easier to understand than the system itself.

Many trust models and trust management protocols have been proposed to assist the task of establishing trust between two communicating peers. There has, however, been no research on developing a pictorial language to model the trust relationship between two peers. We propose a pictorial language for modeling the trust relationship between two peers. We proposed the use of Trust Class Diagrams, Trust Case Diagrams and Trust Transition Diagrams to model the various attributes of the trust relationships and this work is given in reference [36].

In order to model trust between two peers, we introduce the concept of trust tuple. We define a trust tuple as: ‘An ordered set of attributes of a trust relationship that conveys information about the trust relationship’. We propose a generic form of the trust tuple along with the order of the trust attributes, and this is as follows: [Trusting Peer, Trusted Peer, Context, Type of Trust, Trustworthiness, Start Time, End Time].

Use Case: We make use of the use case symbol, for depicting the trust tuple associated with the trust relationship. We define a trust use case as *a use case that is used to symbolize the trust tuple of a given trust relationship*. Finally, we define a trust case diagram as *an extension to the existing use case diagram in UML that can model the trust relationships*. Applying the above modified modeling notations to the above example of Napster, we get the following Trust Case for the trust relationship between the two peers, Alice and Bob.



**Fig. 1.** Trust Case Diagrams for peer-to-peer trust modeling

We propose the use of use of Trust Class Diagrams to model the attributes of the trust relationships between two peers along with the attributes of the peers involved in the trust relationship. For modeling trust relationships between two peers we need just two notations from the class diagrams in UML, namely class and association relationship. We propose slight modifications to the existing UML semantics of class and the *association relationships* [35].

A peer may have a trust relationship with more than one peer. Similarly, a trusting peer may have multiple trust relationships with the trusted peer each in a different context. In this section, we show how trust relationships with multiple (more than one) peers can be modeled using the trust modeling diagrams. Special care needs to be taken to model these as shown in ref [35].

## 7 Factors Influencing Trust and Trust Relationships in Peer-to-Peer (P2P) Systems

Trust in simple terms can be defined as the confidence that the trusting peer has in the trusted peer's ability and willingness to do something in a way that it wishes it to do.

As is evident from the above description, this confidence that the trusting peer has in the trusted peer, would lead to a bond or link between the two peers. We term this link that is created between two peers due to the trust that one has in the other as *trust relationship*.

We defined a trust relationship as '*a bond or association between the involved peer/s, which signifies the trust between the involved peer/s*'.

In other words trust relationship between two peers is an outcome of the trust that one peer has in the other. Hence we feel that all the characteristics of trust like, context dependent nature of trust, dynamic nature of trust, transitive nature of trust, asymmetric nature of trust and event based nature of trust would be the characteristics of trust relationships as well.

We classify a rating that communicates the intensity of the trust relationship as the *trustworthiness of the trust relationship*. This rating conveys how strong the trust relationship between the two peers is.

In (Hussain, Chang and Dillon, 2004) we define trustworthiness as "as a numeric value that depicts the level of trust between two peers at a given time period in a given context and depends upon the intrinsic type of peers in a trusted relationship".

Factors that can communicate trust to the trusting peer in peer-to-peer e-commerce can be catalogued into three, namely:

1. Pre-interaction Factors
2. Reputation Factors
3. Personal Interaction Factors

We define Pre-interaction Factors as 'those factors which can influence the trusting peer, whether to trust the trusted peer, before any interaction between the trusting peer and the trusted peer takes place'. We identify the following three factors in this catalogue:

- i. Psychological nature of the trusting peer
- ii. Attitude of the trusting peer towards P2P e-commerce
- iii. Previous interactions with the trusted peer

As we mentioned previously in P2P communications, there is no central authority to enforce trust-inducing mechanisms like in the client-server environment. As a result, the trusting peer, in order to decide whether to trust the trusted peer, asks other accessible peers about the trustworthiness of the intended peer. These other peers communicate an indication of the trustworthiness of the trusted peer. This method of asking the other peers in the network about the trustworthiness of the trusted peer can help the trusting peer in deciding whether it should trust the trusted peer. We call this gathered information on the trustworthiness of the trusted peer its 'Reputation'. We define reputation factors 'as those factors pertaining to the reputation of the trusted peer and can influence the trusting peer in deciding whether to trust the other peer or

not'. We identify four major factors pertaining to the reputation of the trusted peer, which can influence the decision of the trusting peer:

- iv. Trusted Reputation
- v. Unknown Reputation
- vi. Positive Reputation
- vii. Negative Reputation

We define Personal Interaction Factors 'as those factors which help the trusting peer to associate a trustworthiness value to the trusted peer based on its personal interaction with the trusted peer'. Based on these personal interaction factors, the trusting peer can assign a specific trust value to the trusted peer and decide whether to trust the trusted peer in the future.

We identify two main factors in this category. They are:

- vi. Expected Behavior
- vii. Correlation

## 8 Conclusion

In this paper we carried out a definition of trust, reputation, and proposed a framework for trust modelling, and a trust protocol. We utilize this to develop a trusted environment for virtual collaboration in our future work.

## References

1. Hussain, F., Chang, E. & Dillon, T. S., 2004, 'Classification of Trust in Logistic Peer-to-Peer Communication', Proc. of the Int. Conf: Sciences of Electronic, Technologies of Information and Telecommunications (Accepted Paper, To Appear).
2. Dragovic, B., Kotsovinos, E., Hand, S., Pietzuch, P. (2003), 'Xeno Trust: Event Based distributed trust management', in DEXA, 1<sup>st</sup> ed, IEEE, Los Alamitos, California, Prague, Czech Republic, pp. 410-414.
3. Ratnasingham, P., 'The Importance of Trust in electronic commerce', Electronic Networking Applications and Policy, vol. 8, pp. 313-321.
4. G. Mallach, E., 2000, Decision and Data warehouse Systems, Irwin Mc Graw Hill Companies.
5. Gupta, M., Judge, P., & Ammar, M. "A Reputation System for Peer-to-Peer Networks". In ACM 13th Int. Wksp on Network & Oper. Sys. Support for Digital Audio & Video (NOSSDAV 2003).
6. Chan, H., Lee, R., Dillon, T. S. & Chang, E., E-Commerce, 1 edition, John Wiley and Sons, Ltd.
7. Rahman, A., Hailes, S., Supporting Trust in Virtual Communities, Available: [<http://citeseer.nj.nec.com/cache/papers/cs/10496/http:zSzzSzwww-dept.cs.ucl.ac.ukzSzcgibinzSzstaffzSzF.AbdulRahmanzSzpapers.plzQzhicss33.pdf/abdul-rahman00supporting.pdf>].

8. Gambetta, D., (1990), Can we trust trust? Available:  
[<http://www.sociology.ox.ac.uk/papers/gambetta213-237.pdf>] (4/09/2003).
9. McKnight 1996, The meaning of Trust, Available:  
[<http://miscr.umn.edu/wpaper/WorkingPapers/9601.pdf>] (4/09/2003).
10. Marsh, S (1994), Formalizing Trust as a Computational Concept, Ph.D., University of Sterling.
11. Wang, Y., Vassileva J, Trust and Reputation Model in Peer-to-Peer, Available:  
[[www.cs.usask.ca/grads/yaw181/publications/120\\_wang\\_y.pdf](http://www.cs.usask.ca/grads/yaw181/publications/120_wang_y.pdf)] (15/10/2003).
12. Singh, A. Liu, L., TrustMe: Anonymous Management of Trust Relationships in Decentralized P2P systems, Available:  
[<http://www.cc.gatech.edu/~aameek/publications/trustme-p2p03.pdf>] (11/10/2003).
13. Ooi, Beng C., Liau, Chu Y., Tan, Kian-Lee, Managing Trust in Peer-to-Peer Systems Using Reputation-Based Techniques, Available:  
[<http://citeseer.nj.nec.com/cache/papers/cs/30109/http:zSzzSzwww.comp.nus.edu.sg:zSzzSz~oobczSzwww03.pdf/managing-trust-in-peer.pdf>] (11/14/2003).
14. Cornelli, F., Damiani, E., Vimercati, Sabrina De Capitani di Vimercati, Paraboschi, S., Samarati, P. (2003), Choosing Reputable Servents in a P2P Network, Available:  
[<http://citeseer.nj.nec.com/cache/papers/cs/26951/http:zSzzSzseclab.crema.unimi.it:zSzzSzPapersSzwww02.pdf/choosing-reputable-servents-in.pdf>] (20/9/2003).
15. Aberer, K, Despotovic, Z., Managing Trust in a Peer-2-Peer Information System, Available:  
[<http://citeseer.nj.nec.com/aberer01managing.html>] (11/9/2003).
16. Xiong, L., Liu, L., A Reputation-Based Trust Model for Peer-to-Peer eCommerce Communities, Available:  
[<http://citeseer.nj.nec.com/xiong03reputationbased.html>] (9/10/2003).
17. Chen, R., Yeager, W, Poblano: A distributed Trust Model for Peer-to-Peer Networks, Available: [<http://www.jxta.org/docs/trust.pdf>] (20/9/2003).
18. Kamvar, Sepandar D., Schlosser, Mario T., Garcia-Molina, Hector, The EigenRep Algorithm for Reputation Management in P2P Networks, Available:  
[<http://citeseer.nj.nec.com/kamvar03eigentrust.html>] (11/9/2003).
19. <http://freenet.sourceforge.net/>
20. <http://www.napster.com/>
21. <http://www.jxta.org/>
22. <http://www.gnutella.com/>
23. <http://web.icq.com/>
24. <http://messenger.msn.com/>
25. <http://freenet.sourceforge.net/>
26. Yu, B., Singh, M. P., (2002). "Distributed Reputation Management for Electronic Commerce." *Computational Intelligence* 18 (4): 535-549.
27. Hussain, F., Chang, E. & Dillon, T.S., 2004, 'Classification of trust in peer-to-peer (P2P) communication', *Int. Journal of Computer Science Systems and Engineering* March 2004, Volume 19(2); 59-72.
28. Hussain, F., Chang, E. & Dillon, T.S., 2004, 'Classification of trust in peer-to-peer communication', *Proc. of the IEEE International Conference on Sciences of Electronic, Technologies of Information and Telecommunications (SETIT 2004)*, Tousse, Tunisia. 15-20 March 2004
29. Hussain, F., Chang, E. & Dillon, T.S., 2004, 'Factors Influencing Trust and Trust Relationships in Peer-to-Peer Based e-commerce', *Proc. of 13th EICAR (European Institute of Computer Anti-virus research) Conference*. Luxembourg, 1-4 May 2004.



30. Hussain, F., Chang, E. & Dillon, T.S., 2004, 'Defining Trust in Peer-to-Peer Communication', Accepted by Fourth IEEE and ACM conference on International Network Conference (INC 2004), Plymouth, UK. 6-9 July 2004.
31. Hussain, F., Chang, E. & Dillon, T.S., 2004, 'Peer to Peer (P2P) Trust Modelling Using Trust Case Diagrams', Proceedings of the International Conference on Communication and Broadband Networking (ICBN2004), Kobe, Japan. (accepted for April 7-9, 2004).
32. Hussain, F., Chang, E. & Dillon, T.S., 2004, 'Repute Relationships for Reputation Management in Peer-to-Peer Communication--I', Proceedings of the 2004 International Conference on Security and Management (SAM 2004) in the 2004 International Multi Conference in Computer Science & Computer, Las Vegas (accepted for June 21 - 24, 2004 ).
33. Chang, E., Gardner, W., Talevski, A. & Kapnoullas, T., 2004, A Virtual Logistics network and an E-Hub as a Competitive Approach for Small to Medium Size Companies, Available: [[http://www-staff.it.uts.edu.au/~rajugan/publication/virtual\\_logistics.pdf](http://www-staff.it.uts.edu.au/~rajugan/publication/virtual_logistics.pdf)].
34. Chang, E., Talevski, A., Dillon, T., (2003). Web Service Integration in the Extended Logistics Enterprise. IEEE Conference on Industrial Informatics, Banff, Canada. [http://www-staff.it.uts.edu.au/~rajugan/publication/virtual\\_logistics.pdf](http://www-staff.it.uts.edu.au/~rajugan/publication/virtual_logistics.pdf)
35. Hussain, F.K, Chang, E., Dillon, T., "Peer to Peer Trust Modeling" Proceedings of 13th EICAR (European Institute of Computer Anti-virus research) Conference. Luxembourg, 1-4 May 2004.

# Future Interconnection Environment – Dream, Principle, Challenge and Practice

## Keynote

Hai Zhuge

China Knowledge Grid Research Group  
Key Lab of Intelligent Information Processing  
Institute of Computing Technology, Chinese Academy of Sciences  
zhuge@ict.ac.cn

**Abstract.** Networks and flows are everywhere in society, nature and virtual world organizing versatile resources and behaviors. Breaking boundaries, this keynote establishes a scenario of the future interconnection environment – a large-scale, autonomous, live, sustainable and intelligent network environment where society, nature, and the environment harmoniously co-exist, cooperatively work and evolve. It automatically senses and collects useful information from the nature, transforms raw information into resources in the environment, and organizes resources in semantic rich normal forms that can be easily understood by both machine and human. Geographically dispersed people and resources can work in cooperation to accomplish tasks and solve problems by actively participating material flows, energy flows, technology flows, information flows, knowledge flows and service flows in the environment through roles and machines, improving the environment and the nature. Parameters, characteristics and principles of the environment are proposed from the system methodology point of view. A case study on a live semantic link network under a reference model of the environment is presented. As a practice, the advanced architecture and technology used in China Knowledge Grid e-science environment IMAGINE-1 is introduced.

## 1 Introduction – Dream

Jim Gray summarized the computing history in his 1999 Turing Award Lecture: the dream of Charles Babbage has been largely realized, the dream of Vannevar Bush has almost become reality [5], but it is still difficult for computer systems to pass the Turing Test – computing systems still do not have human intelligence although significant progresses have been made. He extended Babbage's computational goal to include highly secure, highly available, self-programming, self-managing, and self-replicating characteristics [8].

Scientists have made significant progresses in establishing highly secure and highly available systems. But so far, we are still far from the goal of self-programming, self-managing and self-replicating.

Jim Gray extended Vannevar Bush's Memex vision to an ideal that automatically organizes, indexes, digests, evaluates, and summarizes information (scientists in information processing area have been pursuing this goal). He proposed three more Turing Tests: prosthetic hearing, speech, and vision.

The Internet interconnects globally distributed computers and enables file transmission. The World Wide Web interconnects human readable Web pages and enables an easy utility mode. The Semantic Web is to establish common understandings between Web resources by establishing ontology and logical mechanisms, and using standard markup languages like RDF (Resource Description Framework). The Web Intelligence is to improve the current Web by using artificial intelligence technologies and information processing technologies such as symbolic reasoning, text mining, information extraction and information retrieval. The Web Service aims at providing an open platform for the development, deployment, interaction, and management of globally distributed e-services based on Web standards. It enables the integration of services residing and running in different sites.

The global Grid (<http://www.gridforum.org>) aims at sharing, managing, coordinating, and controlling distributed computing resources, which could be ideally machines, networks, data, and any types of devices. The ideal of the Grid is that any compatible devices could be plugged in anywhere onto the Grid and be guaranteed the required services regardless of their locations, just as the power grid. The Grid computing almost excludes previous Web technologies. The Semantic Grid (<http://www.semanticgrid.org>) attempts to incorporate the advantages of the Grid, Semantic Web and Web Services. The Grid architecture has shifted to the service-oriented Open Grid Services Architecture (OGSA), where we can see some features of Web Services [7]. Incorporating Grid with the Peer-to-Peer (P2P) technology through standard mechanisms enables autonomous computing objects to work in a scalable network [2]. P2P networks widely exist in one form or another in society. Incorporating contents such as semantics and trust into the P2P network bridges the existing gap between the P2P network and high-level intelligent applications.

Our modern society requires future interconnection environments to go far beyond the scope of the Turing Test and other automatic machine intelligence problems such as self-programming as well as the current Web/Grid technologies, because the computing environment has evolved from personal or centralized computers to distributed network, to human-computer environments, and to large-scale human-machine environments, where dynamics, evolution, cooperation, fusion, sustainability, and social issues in computing have become major concerns.

## 2 Principle

As a large-scale dynamic human-machine system, a future interconnection environment includes the following five major parameters:

1. *Space* – the carrier of versatile resources as individual or community. Resources are sharable objective existence in the environment such as material, information, knowledge, services, the space, and even the environment.
2. *Time* – the arrow of evolution and degeneration.
3. *Structure* – the construction of the environment and resources.
4. *Relation* – relationships among parameters and among resources.
5. *Viewpoint of value* – the evaluation of state and the evolution trend of resources and their relationships.

Knowledge in nature is the product of society. It evolves and endures throughout the life of a race rather than that of an individual [5]. The future interconnection environment is a growing environment, where knowledge grows through social activities at different levels (from low object layer to high human-machine community layer) and in different discipline spaces.

Human's social activities generate and develop the semantics of natural languages. The semantics in the environment is a kind of human-machine semantics, which establishes common understanding between human and resources. Services, knowledge and regulations exist and work in such semantics to support intelligent activities.

As a complex system, the future interconnection environment develops with the following principles.

1. *Open principle*. To avoid equilibrium, the environment should be open. Standards are important for open systems. The evolution characteristic of the environment requires the standards to be adaptive.
2. *Incremental development principle*. The environment develops with the growth of developers and users who develop, use and update – from small-scale at the early stage to large-scale and at the exponential growth stage. From the perspective of applications, the development of the future interconnection environment should make the balance between inheritance and innovation. It should absorb the advantages of the current Web, Semantic Web, distributed systems, Grid and P2P computing [2, 4, 6, 7].
3. *Economy principle*. Participants, resources and the environment should benefit each other under reasonable anticipation and in fair ways. Economics concerns three objects: participants, the market and the economic system. The market is an important mechanism for automatically and reasonably adjusting market participants' decisions and behaviors. The market participants including producers and consumers pursue the satisfied rather than the optimized exchange under agreement constraints. Based on simple principles, the market mechanism realizes adaptation by avoiding complex computation.
4. *Ecology principle*. Ecology is a study of nature, human society and economy. In future, human society will be involved in an organic interconnection environment that actively interacts with the nature. The principles of nature ecology implicate the formation of the ecology for future interconnection environment [10].
5. *Competition and cooperation principle*. Resources compete for survival, right and reputation in the environment. On the other hand, they cooperate with each other under some conditions (e.g., interest) and regulations to form integrated functions and services.

6. *Dynamic scalability principle*. Participants and resources can join or leave the environment without affecting the overall function of the running environment. The dynamic scalability needs the support from the network layer, the relation (semantic) layer and the organization layer.
7. *Integrity and uniformity principles*. The integrity requires us to guarantee the correctness, and the uniformity requires us to guarantee the simplicity when we develop the future interconnection environment.

### 3 Challenge

The following issues challenge the implementation of the ideal interconnection environment:

1. *Normal re-organization*. The theory that organizes resources in semantic rich normal forms and operates resources under integrity constraints.
2. *Semantic interconnection*. The approach that enables versatile resources to be interconnected in multiple semantic layers to achieve consistency in supporting intelligent applications.
3. *Intelligent clustering and fusing*. The approach enables versatile resources to dynamically cluster and fuse to provide intelligent services. The key is to develop a unified resource model. Established in 2001, China Knowledge Grid Research Group (<http://kg.ict.ac.cn>) has been pursuing the best solutions to above three key issues and has achieved significant progresses (<http://www.knowledgetgrid.net>).
4. *Dynamic inheritance*. Inheritance is a common phenomenon in the organism world and it is also the key mechanism in the Object-Oriented methodology and systems for realizing reuse. How to realize the inheritance mechanism among evolving resources in an evolving environment is a challenging issue.
5. *Degeneration rule of network*. Researchers have studied the growth and distribution of the World Wide Web [1], but they have neglected the degeneration phenomenon. It is common in real world that the development of anything has a limitation. Even for a real-world scale-free network, “the rich gets richer” rule is not true forever in its evolution process. What is the evolution mode under the limitation? Is there any degeneration rule of networks? Solution to this issue will form significant contribution to the theory of the future interconnection environment.
6. *Abstract flow model*. This model unveils the common rules and principles of material flow, information flow, knowledge flow and service flow, and implements flow logistics. It concerns the abstract process model and control mechanism.
7. *Field theory of interconnection environment*. Just as in physical world, versatile resources in the interconnection environment constitute a special field. Resources flow from high power nodes to low power nodes. However, the duplication and generation of resources do not cause the loss of any other resources such as energy in the physical world, and the loss of power also does not lead to the loss of any other materials such as energy. The law of energy conservation does not hold in this special field. Unveiling the laws and principles in this field significantly contributes to the theory of the future interconnection environment.

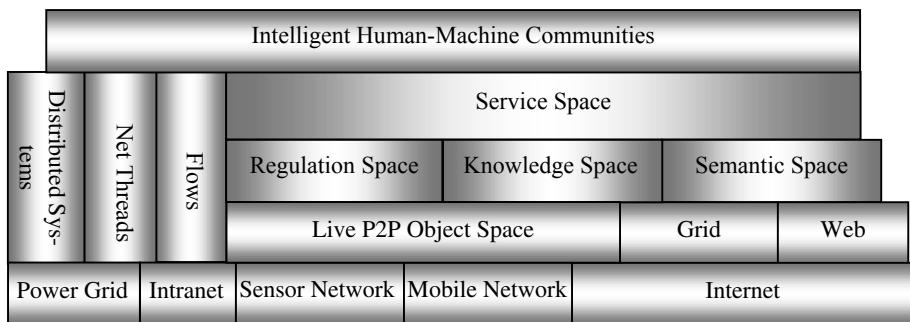
8. *Abstraction and reasoning in Single Semantic Image.* The Single Semantic Image (SSeI) establishes common understanding for versatile resources [9]. It is a challenging issue to automatically capture semantics from versatile resources, to make abstraction, and to reason and explain in a uniform semantic space. Semantic constraints and related rules for SSeI guarantee the correctness of resource operations at the semantic layer.
9. *Interconnection environment ecology.* Resources in the interconnection environment can be classified into different species. A resource can be generated only by inheriting from existing species. The evolution of species depends on flows between species. The methods and principles of natural ecology help us to investigate the interconnection environment ecology.
10. *Methodology.* The large-scale, dynamic and intelligent interconnection environment challenges previous software and system methodology. The new methodology includes a multi-disciplinary system methodology and an epistemology for guiding the development, operation and maintenance of the future interconnection environment and applications.

Due to the variety and complexity of the environment, any unary theory is limited in their ability to support the modeling of the future interconnection environment. Breaking the boundaries of disciplines often builds insights to solve these challenging issues.

## 4 Case Study

### 4.1 Reference Architecture

According to the incremental development principle, we propose the layered reference architecture as shown in Fig. 1.



**Fig. 1.** Reference architecture

The live P2P object space is the abstraction of versatile resources in the environment and each object has a life span from birth to death. The net threads realize run-time applications. Regulations in the regulation space are for the autonomous man-

agement of resources. The knowledge space and the semantic space form content overlays on the live P2P object space, and they support services in the service space. Services can actively find requirements advertised by roles and resources. Human and roles in the intelligent human-machine communities work, entertain, contribute services, and enjoy services in the environment according to regulations and social principles. A role can move from one community to another through definite corridors, which establish logistics between communities.

## 4.2 Explore the Live Semantic Network

A live semantic link network consists of live resource nodes and semantic links between nodes. Any node has a life span from birth – adding it to the network – to death – removing it from the network. We investigate the evolution rules of such a network by considering both the adding and removing behaviors of nodes and links. Previous research focuses on the adding behavior. Through establishing and investigating two types of models: a stochastic growth model and a directed evolving graph model, we obtain the same scale-free distribution rule.

We make a comparison between the current hyperlink distribution and the live semantic link distribution. Fig.2 (a) and (b) show the in-degree distribution and the out-degree distribution. We can clearly see the offset between the current Web model and our model. The key reason lies in the link and node delete operations in our model. Obviously, the bigger the delete value is, the larger the offset there may be. In addition, the absolute value of the slope in the curve for our model is always smaller than that of the current hyperlink network.

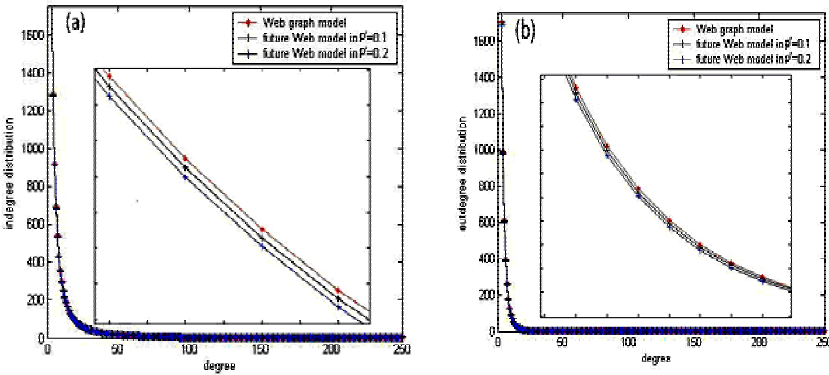
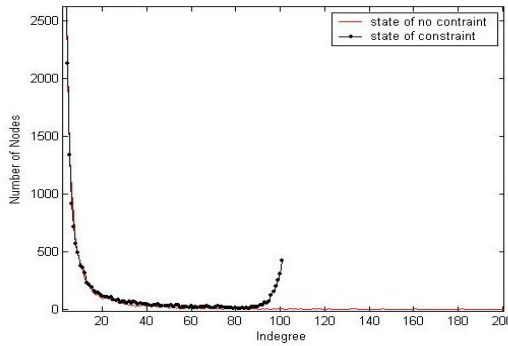


Fig. 2. Comparison between previous model and our model

Nodes' wealth can be evaluated by the number of links they possess. Richer nodes own larger number of links. The preferential attachment mechanism leads to a 'rich get richer' phenomenon [3]. This means the gap between the rich and the poor will be enlarged unlimitedly. Is it true under some constraints?

Real cases tell us, the rich will endure more for being richer than the poor. A firm limitation leads to average the wealth. Our simulation carries out by setting the following constraint: if the number of in-links in one node reaches a certain amount, subsequent links will be no longer attached to it. The simulation result shown in Fig. 3 tells us the distribution is no longer a power law. Its tail rises a little near the limitation. This observation indicates that wealth has been shared among relatively rich nodes.



**Fig. 3.** Power law changes under constraints

## 5 Practice in e-Science

China e-Science Knowledge Grid Environment IMAGINE-1 is our practice to realize the ideal of the future interconnection environment. IMAGINE-1 aims at an evolving, self-organized, self-managed and large-scale human-machine environment. To build such a complex system, we have solved the following issues:

1. The semantic-based organization model of resources on P2P networks.
2. Decentralized resource monitoring, scheduling and management that are embedded in applications.
3. The advanced architecture and application framework supporting the development of high-level applications.

Its architecture has the following three distinguished characteristics:

1. It is a general and scalable running platform that supports the development of domain applications.
2. It is a virtual experiment environment to simultaneously support research and development of the environment and research and development of domain applications.
3. It integrates the research and development laboratory with the real application running platform so that experimental applications can be tested in a real environment with users' participation.



IMAGINE-1 implements an embedded resource management platform on a P2P network to support self-organization and self-management of applications. Sophisticated applications can be developed to provide scalable services on large-scale networks based on IMAGINE-1.

IMAGINE-1 consists of the following platforms:

1. IMAGINE-Framework, a basic application development framework that implements a layered architecture to provide the low-level resource organization of a P2P network, the embedded resource management for high-level applications, and the distributed architecture implementation for developing high-level applications.
2. IMAGINE-Run, a running platform that supports the run-time management of the underlying P2P network and high-level applications, including network maintenance, application deployment, configuration, and running control.
3. IMAGINE-Research, a virtual network laboratory that supports the monitoring, debugging, configuring, testing and verifying of research for improving the environment's technologies.
4. IMAGINE-Builder, a platform for efficiently developing distributed domain applications based on a virtual component technology.
5. EcoLab, a distributed scientific research laboratory for ecologists.

Fig. 4 depicts the IMAGINE-1 organism. Its construction process is as follows:

1. Researchers develop the IMAGINE-Framework to implement the organization and management of resources in decentralized and autonomous way on a P2P network. It is the core component that implements the environment.
2. Use the IMAGINE-Framework to support the development of the IMAGINE-Run that implements a distributed network environment supporting P2P network management, application deployment, running management and user management.
3. Use the IMAGINE-Framework to support the development of the IMAGINE-Research as a P2P-based virtual laboratory that supports research on the environment. The IMAGINE-Research enables researchers and users to fully interact with each other to form a positive feedback cycle of requirements and technologies that pushes the evolution of the environment.
4. Use the IMAGINE-Framework to support the development of the IMAGINE-Builder, which consists of building tools for component specification, component building, and component-based applications. The IMAGINE-Builder can facilitate distributed application development on large-scale networks based on distributed services reuse and integration.
5. Develop the EcoLab on the IMAGINE-1. EcoLab is an e-science Knowledge Grid that supports geographically dispersed knowledge sharing, integration and utilization, and supports coordinated scientific research on a large-scale P2P network.

When IMAGINE-Run and IMAGINE-Research are developed and deployed, they can be used to improve and upgrade the IMAGINE-Framework on large-scale networks. This forms a beneficial technology evolution cycle for the development of IMAGINE-1. The IMAGINE-1 builder enhances the development of domain applications. The EcoLab can in turn feed back users' requirements and can further help improve the whole environment in both the platforms and the domain applications.

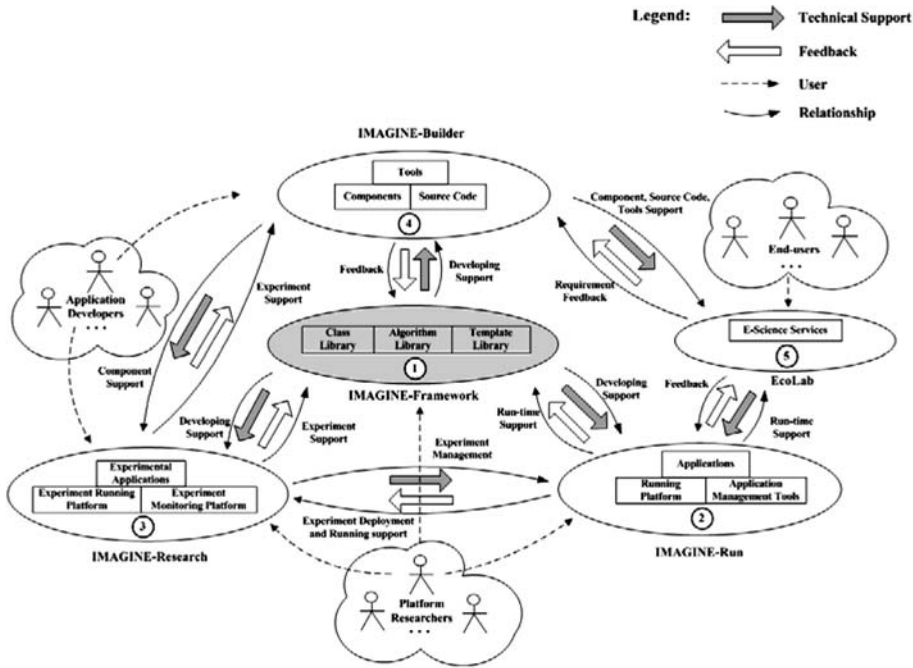


Fig. 4. IMAGINE-1 organism

On the other hand, IMAGINE-Framework, IMAGINE-Research and IMAGINE-Run involve different user communities: the platform researchers, the domain application developers and the end-users. They interact and coordinate with each other forming harmonious development processes and evolution processes of the platforms and the whole environment.

EcoLab supports ecologists in geographically dispersed academic institutions to efficiently and effectively publish, share, manage and utilize distributed resources including computing power, data, information and knowledge.

The distinguished characteristics of IMAGINE-1 facilitate various kinds of distributed applications from the simple data and file sharing to the complex distributed problem solving, from the distributed text-based instant communication to the high-speed multi-media broadcasting, and from the novel decentralized network game to the intelligent information and computing services.

## 6 Summary

Nature, human society, the future interconnection environment and its elements are autotrophic and harmonious symbiosis organisms. They cooperatively work, evolve and develop according to the open, incremental development, economy, ecology, competition and cooperation, dynamic scalability, integrity and uniformity principles.

The environment pursues a sustainable and harmonious system function of space, time, structure, relation, and viewpoint of value.

China Knowledge Grid Research Group leads a cooperative team working on the National Semantic Grid Plan of China, a five-year research plan supported by the Ministry of Science and Technology of China. Nine universities in China participate in its research. The applications concern e-science, e-culture, weather service and public health.

## Acknowledgement

The research work is supported by the National Grand Fundamental Research 973 Program and the National Science Foundation of China. The author thanks Xiaoping Sun, Jie Liu, Xiang Li, and Xue Chen for their help in simulation.

## References

1. Adamic, L.A. and Huberman, B.A.: Power-Law Distribution of the World Wide Web. *Science*. 287 (24) (2000) 2115.
2. Balakrishnan, H. et al.: Looking Up Data in P2P Systems. *Communications of the ACM*. 46 (2) (2003) 43-48.
3. Barabasi, A.L. and Abert, R.: Emergence of Scaling in Random Networks. *Science*. 286 (1999) 509-512.
4. Berners-Lee, T., Hendler, J., and Lassila, O.: Semantic Web. *Scientific American*. 284 (5) (2001) 34-43.
5. Bush, V.: As We May Think. *The Atlantic Monthly*. 176 (1) (1945) 101-108.
6. Cass, S.: A Fountain of Knowledge. *IEEE Spectrum*. 41(1) (2004) 60-67.
7. Foster, I., et al., Grid Services for Distributed System Integration. *Computer*. 35 (6) (2002) 37-46.
8. Gray, J.: What Next? A Dozen Information-Technology Research Goals. *Journal of the ACM*. 50 (1) (2003) 41-57.
9. Zhuge, H.: China's E-Science Knowledge Grid Environment. *IEEE Intelligent Systems*. 19 (1) (2004) 13-17.
10. Zhuge, H.: Eco-Grid: A Harmoniously Evolved Interconnection Environment. *Communications of the ACM*. Accepted Jan.2003, to be appeared in 2004.

# Recent Advances in Histogram Construction Algorithms

Kyuseok Shim

Seoul National University  
Seoul, Korea  
shim@ee.snu.ac.kr

**Abstract.** Obtaining fast and accurate approximations to data distributions is a problem of central interest to database management. A variety of database applications including, approximate querying, similarity searching and data mining in most application domains, rely on such accurate approximations. A very popular way in database theory and practice to approximate a data distribution is by means of a histogram.

Histograms approximate a data distribution using a fixed amount of space, and under certain assumptions strive to minimize the overall error incurred by the approximation. The problem of histogram construction is of profound importance and has attracted a lot of research attention. A fundamental requirement for any good histogram construction algorithm is to approximate the underlying data distribution in a provably good way and be efficient in terms of running time. The typical assumption for constructing histograms is that the data set to be approximated is finite and of known size, or that the size can be easily derived by performing a single pass over the finite data set. The problem of histogram construction on datasets of known size has received a lot of research attention and the optimal as well as a lot of heuristic solutions exist. Wavelets synopses is in fact very simple histograms, and this approach allows us to minimize some objective fit criterion, than storing the  $k$  highest coefficients of a wavelet decomposition. In all these problems, given a sequence of data values  $x_1, \dots, x_n$ , the task is to construct a suitable summary of the data which can be stored in small space (e.g. a small fixed number, say  $B$ , of the  $n$  coefficients in the Wavelet transform, or a histogram involving  $B$  buckets).

In this talk, we first present optimal and approximate histogram construction algorithms that have been recently proposed. We next discuss the algorithms to approximate a data stream in an online and incremental way using histograms. We also describe wavelet synopses algorithms that include deterministic and probabilistic thresholding schemes to select the wavelet coefficients. We finally point out the drawbacks of such existing schemes and introduce future research directions.

# Dynamic Adjustment of Sliding Windows over Data Streams

Dongdong Zhang<sup>1</sup>, Jianzhong Li<sup>1,2</sup>, Zhaogong Zhang<sup>2</sup>, Weiping Wang<sup>1</sup>,  
and Longjiang Guo<sup>1,2</sup>

<sup>1</sup> School of Computer Science and Technology, Harbin Institute of Technology, China  
{zddhit, lijzh, wpwang, guolongjiang}@hit.edu.cn

<sup>2</sup> School of Computer Science and Technology, Heilongjiang University, China  
zhangzhaogong@hit.edu.cn

**Abstract.** The data stream systems provide sliding windows to preserve the arrival of recent streaming data in order to support continuous queries in real-time. In this paper, we consider the problem of adjusting the buffer size of sliding windows dynamically when the rate of streaming data changes or when queries start or end. Based on the status of available memory resource and the requirement of queries for memory, we propose the corresponding algorithms of adjustment with greedy method and dynamic programming method, which minimize the total error of queries or achieve low memory overhead. The analytical and experimental results show that our algorithms can be applied to the data stream systems efficiently.

## 1 Introduction

In many recent application systems, data takes the form of continuous *data streams*, rather than finite stored data sets. Examples include stock ticks in financial applications, performance measurement in network monitoring and traffic management, log records or click-streams in Web tracking and personalization, data feeds from sensor applications, network packets and messages in firewall-based security, call detail records in telecommunications, and so on. These application systems have rapid and time-varying streams and require timely online answers, which we call as *data stream system* generally. The data stream can be denoted as an unlimited set with the model of  $\langle s, t \rangle$ , where  $s$  denotes a tuple and  $t$  denotes a timestamp which is determined by the time when  $s$  enters the data stream system or when the data source produces  $s$  [10]. Since the memory of the system is limited and streaming data is unlimited, the data stream system provide *sliding windows* for each data stream, which preserve the arrival of recent data in order to support queries in real-time. Differing from the traditional queries, the queries in the data stream systems, named as *continuous queries*, stay in the system with long time. As the streaming data enters the data stream system continuously, the continuous queries accept and operate it, then output the querying-results. There may be more than one data stream in the data stream system. In general, each data stream corresponds to one sliding window and the same sliding window can be accessed by many different continuous queries. Furthermore, one continuous query may query the streaming data over several sliding windows.

At present, there is some work [1–4] about how to deal with the streaming data in the sliding windows. [8] discusses how to analyze whether a query can be supported with the limited memory. [5] considers scheduling the operators optimally in order to save the buffer size between operators. [4] minimizes the response time and maximizes the query throughput by scheduling Joins which share the sliding window in data stream system. Kang et al. [9] discuss the cost model of Join over the unlimited streaming data. All these work is on the assumption that the buffer size of sliding windows is static. However, different continuous queries have different querying-range. Furthermore, in data stream systems, the requirement of queries for memory is dissimilar and the rate of the streaming data changes rapidly. If we fix the buffer size of sliding windows, it will cause the problem that some sliding windows' buffer is vacant and other sliding windows' buffer is too small to support queries. We should break out the assumption about fixing the buffer size in previous work, and adopt a feasible mechanism for adjusting the buffer size of sliding windows to support continuous queries more efficiently.

In this paper, we propose the algorithms of adjusting the buffer size of sliding windows dynamically with greedy method and dynamic programming method, in order to support queries more efficiently with the limited available memory, minimize the total error of queries or achieve low memory overhead. The problem of adjustment is classified into three cases for discussion. We also give analytical and experimental results of the algorithms, which show that our algorithms can be applied to the data stream systems efficiently.

## 2 Problem Definition

### 2.1 Sliding Window

Suppose the streaming data enters the sliding window as the order of timestamp, then, the data in the sliding window at any time can be formed into a series as:

$$\langle s_l, t_l \rangle, \langle s_{l+1}, t_{l+1} \rangle, \dots, \langle s_{u-1}, t_{u-1} \rangle, \langle s_u, t_u \rangle \quad (2-1)$$

where  $t_i \leq t_j$  and  $l \leq i < j \leq u$ . Sliding windows can be defined either in terms of time-unit or tuple-count [1]. In this paper, the algorithms we proposed are based on time-unit sliding windows, in spite that they can also be applied to tuple-count sliding windows in the same way. Assuming  $W$  denotes the set of sliding windows and  $d$  denotes the number of sliding windows in the data stream system, we give Definition 1.

**Definition 1.**  $\forall w \in W$ , supposing the data in  $w$  can be listed as series (2-1), the **win-*down-width*** of  $w$  is defined as  $t_u - t_l$ .

We let  $W_N(w)$  denote the window-width of  $w$  before adjusting, and let  $W_Y(w)$  denote the window-width of  $w$  after adjusting. The timestamp over  $w$  ranges from  $Now - W_N(w)$  to  $Now$ , where **Now** is the current time (system clock). Let **Size**( $w$ ) denote tuple size and let **Rate**( $w$ ) denote the average rate of streaming data over  $w$ , then we can evaluate the buffer size of  $w$  with the window-width of  $W_N(w)$ , which is equal to  $W_N(w) \cdot \text{Size}(w) \cdot \text{Rate}(w)$ . Supposing the total available memory, named as **available-memory**, allocated to all sliding windows is  $M$ , then both  $M \geq \sum_{w \in W} W_N(w) \cdot \text{Size}(w) \cdot \text{Rate}(w)$  and  $M \geq \sum_{w \in W} W_Y(w) \cdot \text{Size}(w) \cdot \text{Rate}(w)$  should hold at any time.

## 2.2 Continuous Query

Following is an example of continuous queries like SQL.

```

SELECT   AVG (temperature)
FROM      $w_1$  [RANGE Now - 100, Now - 20]
WHERE    predicate
ERROR (2%) EVERY (5) BEGIN (70) END (500)
  
```

(2-2)

Let  $Q$  denote the set of continuous queries with the similar syntax with the query (2-2). Generally,  $\forall q \in Q$ , we can extract the *metadata* from the syntax structure of  $q$ .

The querying-object of the query (2-2) is the sliding window  $w_1$ . If the sliding window  $w$  is the querying-object of  $q$ , let  $U(w, q) = 1$ . Otherwise, let  $U(w, q) = 0$ . We call the value of  $Now - (Now - 100) = 100$  as *querying-width* of query (2-2), denoted by  $Q_t(q)$ . Meanwhile,  $\forall w \in W$ , we let  $Max\_T(w)$  denote the maximal querying-width over  $w$ , i.e.  $Max\_T(w) = \max\{Q_t(q) \mid q \in Q \wedge U(w, q) = 1\}$ .

The tolerated error of  $q$  can be extract from the **ERROR** clause, which is either relative error or absolute error. In our work, by the mapping function  $f_q$ , we assume that the tolerated error can be mapped into a value of window-width, denoted by  $Q_e(q)$ . That is, the window-width must be larger than  $Q_t(q) - Q_e(q)$  so that  $q$  can yield the querying-result with the tolerated error at any time. However, the constructing of  $f_q$  is a complexity problem, which will not be discussed here and will be our future work. Fortunately, [6][7] had given some research result about it and may be used to solve the problem. Specifically, if we couldn't construct  $f_q$  directly, such as in case of MAX/MIN queries, we can set  $Q_e(q)$  to zero (that is, the tolerated error of  $q$  is zero), which doesn't impact on our work.

The tolerated delay of yielding the querying-result can be extracted from the **EVERY** clause in  $q$ , denoted by  $Q_d(q)$ . For example, the time interval should not exceed 5 seconds between any two adjacent instant of yielding the querying-result of query (2-2). Finally, we can compute the running duration of  $q$  from the **BEGIN** clause and the **END** clause.

**Definition 2.**  $\forall w \in W$ , let  $W_Y(w)^{(i)}$  denote the window-width of  $w$  at the time  $i$ . Supposing  $\exists q \in Q$  and  $U(w, q) = 1$ .  $\forall i$ , if  $W_Y(w)^{(i)} \geq Q_t(q)$  holds, we call that  $w$  can **A-rank-support**  $q$ ;  $\forall i$ , if  $W_Y(w)^{(i)} \geq Q_t(q) - Q_e(q)$  holds, we call that  $w$  can **B-rank-support**  $q$ ; There is a time series  $t_1, t_2, t_3, \dots$ , where  $t_1 < t_2 < t_3 < \dots$  and  $t_{k+1} - t_k \leq Q_d(q)$  hold for any  $k$ .  $\forall k$ , if  $W_Y(w)^{(t_k)} \geq Q_t(q) - Q_e(q)$  and  $W_Y(w)^{(t_{k+1})} \geq Q_t(q) - Q_e(q)$  hold, we call that  $w$  can **C-rank-support**  $q$ .

A-rank-supporting means that  $w$  can support  $q$  to yield the result without any error at any time. B-rank-supporting means that  $w$  can support  $q$  to yield the result with the tolerated error at any time. C-rank-supporting means that  $w$  can support  $q$  to yield the result with the tolerated error within the tolerated delay. Obviously, if  $w$  can A-rank-support  $q$ ,  $w$  can either B-rank-support or C-rank-support  $q$ . In a similar way, if  $w$  can B-rank-support  $q$ ,  $w$  can C-rank-support  $q$ .

**Definition 3.**  $\forall w \in W$ , we define the value of  $\max\{Q_{t-e}(q) \mid q \in Q \wedge U(w, q) = 1 \wedge Q_{t-e}(q) = Q_t(q) - Q_e(q)\}$  as *min-valid-width* of  $w$ , denoted by  $Min\_T(w)$ .

Definition 3 describes that the window-width of the sliding window  $w$  must be larger than  $Min\_T(w)$  so that every query over  $w$  can be B-rank-supported.

### 2.3 Problem Classification

We focus our attention on adjusting the window-width of sliding windows in real-time based on the dynamic environment. According to the comparison between the available-memory and the requirement of queries for memory, we classify the problem into following three cases for discussion.

Case (1): When  $M \geq \sum_{w \in W} \text{Max\_}T(w) \cdot \text{Size}(w) \cdot \text{Rate}(w)$  holds, the available-memory is rich and can A-rank-support all continuous queries. So, the aim of adjusting the window-width is to support more future queries by the predictive method according to the metadata of registered queries.

Case (2): When  $\sum_{w \in W} \text{Min\_}T(w) \cdot \text{Size}(w) \cdot \text{Rate}(w) \leq M < \sum_{w \in W} \text{Max\_}T(w) \cdot \text{Size}(w) \cdot \text{Rate}(w)$  holds. The available-memory can B-rank-support all continuous queries, rather than A-rank-support. So, the aim of adjusting the window-width is to minimize the total error of the results of all queries.

Case (3):  $M < \sum_{w \in W} \text{Min\_}T(w) \cdot \text{Size}(w) \cdot \text{Rate}(w)$  holds. The available-memory is too small to B-rank-support all queries. So, the aim of adjusting window-width is to C-rank-support all queries.

## 3 Algorithms for Adjusting the Buffer Size

### 3.1 A-Rank-Adjusting Algorithm

When the available-memory is rich, we need adjust the window-width of windows in advance to support more future queries with the predictive method based on the analysis of the metadata of the registered continuous queries, in condition that every query can be A-rank-supported. There is an intuitional idea that for any sliding window  $u$  and  $v$ , if  $\text{Max\_}T(u) > \text{Max\_}T(v)$  holds, we maybe know the requirement of queries for window-width over  $u$  is more great than that over  $v$ . So, for any future query  $q$ , if  $Q_i(q) > \text{Max\_}T(u) > \text{Max\_}T(v)$  holds, in general, the probability of  $u$  being the querying-object of  $q$  is larger than that of  $v$  being the querying-object of  $q$ . Based on this idea, the strategy of adjustment is to reallocate total window-vacant-memory to sliding windows in terms of different ratio of maximal querying-width over sliding windows. Due to the limited space, we omit the description of corresponding algorithm.

### 3.2 B-Rank-Adjusting Algorithm

**Definition 4.**  $\forall w \in W$ , the value of  $\sum_{q \in Q} P(w, q) \cdot [Q_i(q) - W_N(w)]$  is defined as *window-accumulate-error* over  $w$ , denoted by  $E(w)$ , where  $P(w, q)$  is set to 1 when  $U(w, q) = 1$  and  $W_N(w) < Q_i(q)$  hold, otherwise  $P(w, q) = 0$ .

We use window-accumulate-error to measure the total error of querying-result over each sliding window.

When  $\sum_{w \in W} \text{Min\_}T(w) \cdot \text{Size}(w) \cdot \text{Rate}(w) \leq M < \sum_{w \in W} \text{Max\_}T(w) \cdot \text{Size}(w) \cdot \text{Rate}(w)$  holds, the limited available-memory can not A-rank-support all queries. Thus, the aim of adjusting window-width is: (1) all queries should be B-rank-supported at least. (2) minimizing the total window-accumulate-error of all queries, i.e. minimizing  $\sum_{w \in W} E(w)$ . In light of above two aims, the process of adjusting window-width can be di-



vided into two phases. Firstly,  $\forall w \in W$ , we set the window-width of  $w$  to min-valid-width, i.e.  $W_Y(w) = \text{Min\_}T(w)$ . Secondly, we reallocate the remain vacant memory of available-memory to sliding windows gradually, so that we can minimize  $\sum_{w \in W} E(w)$ . Obviously, we should focus our optimal work on the second phase.

**Definition 5.**  $\forall w \in W$ , let  $s_w$  denote the instantaneous window-width of  $w$ , then we call the set  $\{s_w | w \in W \wedge s_w \geq 0\}$  as **windows-snapshot**, denoted by  $S$ , and the window-width of  $w$  in  $S$  is denoted by  $W^{(S)}(w)$ . The set  $\{x_w | w \in W \wedge x_w \geq 0\}$  is defined as **windows-increment**, denoted by  $X$ , and the increment of window-width of  $w$  in  $X$  is denoted by  $W^{(X)}(w)$ .

**Definition 6.** For any windows-snapshot  $S$ , we define the triple  $(w, W^{(S)}(w), \Delta x_w)$  as the **atom-increment** over  $S$ , denoted by  $I$ , where  $w \in W$ ,  $\Delta x_w > 0$ , and  $\Delta x_w$  is the increment based on the window-width with  $W^{(S)}(w)$  over  $w$ .

Supposing there is a windows-snapshot  $S$ , an atom-increment  $I = (w, W^{(S)}(w), \Delta x)$  over  $S$ , and a windows-increment  $X$ , then  $S' = S + I$  is the instantaneous state of window-width after increasing the window-width of  $w$  from  $W^{(S)}(w)$  up to  $W^{(S)}(w) + \Delta x$ .  $S'' = S + X = \{W^{(S)}(w) + W^{(X)}(w) | w \in W\}$  is the instantaneous state of window-width after increasing the window-width of each sliding window from  $W^{(S)}(w)$  up to  $W^{(S)}(w) + W^{(X)}(w)$ . In addition, let  $D(S, X)$  denote the total reduced window-accumulate-error after we changes the windows-snapshot  $S$  into  $S + X$ , where  $S$  called as the **basis** of  $X$ . The expression  $X + I$  means increasing the window-width of  $w$  from  $W^{(X)}(w)$  up to  $W^{(X)}(w) + \Delta x$ .

Based on above analysis, the problem of adjusting window-width optimally can be described as follows.

**Input:**  $M, W, Q$  and initial windows-snapshot  $S_1 = \{\text{Min\_}T(w) | w \in W\}$ .

**Output:** The windows-increment  $X = \{\Delta x_w | w \in W\}$ .

**Object function:**  $\max D(S_1, X)$ .

**Subject to:**  $\sum_{w \in W} W_Y^{(X)}(w) \cdot \text{Size}(w) \cdot \text{Rate}(w) \leq M$ .

(3-1)

**Definition 7.** For each sliding window  $w$ , we define the value of  $|C(w, y)| / [\text{Size}(w) \cdot \text{Rate}(w)]$  as **adjustment-gain** of  $w$  with the window-width  $y$ , denoted by  $G(w, y)$ , where  $C(w, y) = \{q | U(w, q) = 1 \wedge Q_i(q) > y\}$ .

The value of adjustment-gain of the sliding window  $w$  explains the size of reduced window-accumulate-error after we allocate one unit of available-memory to  $w$ . Since the querying-width of all queries is discrete, the adjustment-gain is a subsection-function. For each sliding window  $w$ , let  $L_w = \{a_{w1}, \dots, a_{wi}, \dots, a_{wnw}\}$  denote the set of serial ordered subsection-point of  $G(w, y)$  of  $w$  within the region  $[\text{Min\_}T(w), \text{Max\_}T(w)]$ , where  $1 \leq i \leq n_w$ ,  $a_{wi} < a_{wi+1}$ ,  $a_{w1} = \text{Min\_}T(w)$  and  $a_{wnw} = \text{Max\_}T(w)$ . Considering Definition 7, for each sliding window  $w$ , each of subsection-point in  $L_w$  is the querying-width of one continuous query. Meanwhile, the value of  $G(w, y)$  is always constant when  $x$  is between any two adjacent subsection-points, and we obtain  $G(w, y_i) \geq G(w, y_j)$  only if  $y_i \leq y_j$ .

**Definition 8.** Supposing  $S$  is a windows-snapshot.  $\exists u \in W$ , if  $G(u, W^{(S)}(u)) = \max \{G(w, W^{(S)}(w)) | w \in W\}$  holds, we call  $u$  as the **best-window** over  $S$ . If  $\exists i, a_{wi} \in L_u$  and  $a_{wi} \leq W^{(S)}(u) < a_{wi+1}$  hold, we define the atom-increment  $I = (u, W^{(S)}(u), \Delta x_u)$  as **best-atom-increment** over  $S$ , where  $\Delta x_u = a_{wi+1} - W^{(S)}(u)$  and  $\Delta x_u$  is called as **best-increment** over  $S$ .

We use greedy algorithm to solve the problem (3-1). The main idea is to increase the window-width of the best-window by best-increment at any instant of windows-snapshot. Following will give the greedy algorithm of the problem (3-1).

**Input:**  $M, W, Q, \sum_{w \in W} \text{Min\_}T(w) \cdot \text{Size}(w) \cdot \text{Rate}(w) \leq M < \sum_{w \in W} \text{Max\_}T(w) \cdot \text{Size}(w) \cdot \text{Rate}(w)$

**Output:** For each sliding window  $w$ , output  $W_y(w)$ .

**Error-Based()**

- (1) For each sliding window  $w$ , let  $W_y(w) = \text{Min\_}T(w)$ ; /\*queries can be B-rank-supported\*/
- (2) Calculating the remain available-memory  $M_R = M - \sum_{w \in W} \text{Min\_}T(w) \cdot \text{Size}(w) \cdot \text{Rate}(w)$ ;
- (3) For each sliding window  $w$ , produce the set  $L_w$  of the serial subsection-point of adjustment-gain function within the region  $[\text{Min\_}T(w), \text{Max\_}T(w)]$ ;
- (4) Let the initial windows-snapshot  $S_i = \{X_w | w \in W\}$ , let the initial windows-increment  $X = \{x_w | w \in W \wedge x_w = 0\}$  and  $i = 1$ ;
- (5) **while** ( $M_R > 0$ )
- (6) Finding the best-window  $u$ , the best-atom-increment  $I_i$  and best-increment  $b_i$  over  $S_i$ ;
- (7) **if** ( $M_R \geq b_i \cdot \text{Size}(u) \cdot \text{Rate}(u)$ )  $X = X + I_i$ ;
- (8) **else**  $W^{(X)}(u) = W^{(X)}(u) + M_R / [\text{Size}(u) \cdot \text{Rate}(u)]$ ;
- (9)  $M_R = M_R - b_i \cdot \text{Size}(u) \cdot \text{Rate}(u)$ ;  $S_{i+1} = S_i + I_i$ ;  $i = i + 1$ ;
- (10) For each  $w$ , let  $W_y(w) = W_y(w) + W^{(X)}(w)$  and output the final window-width  $W_y(w)$ ;

The total time complexity of Error-Based algorithm is  $O(\max\{d \cdot c_q, c_q \log c_q\})$ , where  $c_q$  denotes the number of continuous queries.

**Theorem 1.** When  $\sum_{w \in W} \text{Min\_}T(w) \cdot \text{Size}(w) \cdot \text{Rate}(w) \leq M < \sum_{w \in W} \text{Max\_}T(w) \cdot \text{Size}(w) \cdot \text{Rate}(w)$  holds, Error-Based algorithm can produce the optimal solution of the problem for adjusting the buffer size of sliding windows.

### 3.3 C-Rank-Adjusting Algorithm

When  $M < \sum_{w \in W} \text{Min\_}T(w) \cdot \text{Size}(w) \cdot \text{Rate}(w)$  holds, the limited available-memory can't B-rank-support all queries. So, we should try to C-rank-support all continuous queries.  $\forall w \in W$ , supposing  $\exists q \in Q, U(w, q) = 1$  and  $Q_d(q) > 0$  hold. Observing that there is the tolerated delay of  $q$  when yielding the querying-result. Therefore, we can adjust the window-width cyclically with time slice, in order to shorten the window-width of  $w$  just after yielding the querying-result and increase the window-width of  $w$  up to min-valid-width just before yielding the querying-result. Thus, we can free part of vacant memory of  $w$  just after  $q$  yielding the querying-result, so that other sliding windows can use this amount of vacant memory. In this way, all continuous queries can be as possibly as C-rank-supported with the limited available-memory.

We realize it is hard to make sure that all queries over the same sliding window can yield the querying-result synchronously. However, we can pick out a query from all queries over the same sliding window, called as *norm-query*, to be the criterion of reference to adjust the window-width dynamically, so that norm-queries can be C-rank-supported and other queries can be B-rank-supported. Therefore, the memory

resource belong to the sliding window  $w$  is divided into two parts, one (named *stable-memory*) is stably occupied by  $w$ , which makes sure that all queries (except the norm-query) over  $w$  can be B-rank-supported, the other is shared with other sliding windows cyclically, which makes sure that the norm-query over  $w$  can be C-rank-supported.

**Definition 9.**  $\forall w \in W, \exists p \in Q$ , if  $U(w, p) = 1$  and  $Q_{t-e}(p) = \text{Min\_}T(w)$  hold,  $p$  is called as the *norm-query* over  $w$ , and  $Q_d(p)$  is the *allocation-period* of  $w$ , denoted by  $T_D(w)$ . If  $\exists q \in Q, U(w, q) = 1$  and  $Q_{t-e}(q) = \max\{Q_{t-e}(r) \mid r \in Q \wedge r \neq p \wedge U(w, r) = 1\}$  hold, then we call the value of  $\min\{Q_{t-e}(p) - Q_{t-e}(q), Q_d(p)\}$  as the *min-valid-delay* of  $w$ , denoted by  $\text{Min\_}D(w)$ ; otherwise,  $\text{Min\_}D(w) = T_D(w)$ . We call  $Q_{t-e}(q) \cdot \text{Size}(w) \cdot \text{Rate}(w)$  as the *stable-memory* over  $w$ , denoted by *Static*( $w$ ). The *delay-memory* of  $w$  is defined as the value of  $\text{Min\_}D(w) \cdot \text{Size}(w) \cdot \text{Rate}(w)$ , denoted by  $R(w)$ .

In order to schedule sliding windows to use memory reasonably, we should divide sliding windows into several groups (each group called as a *cyclic-group*), so that sliding windows in the same group use the vacant memory cyclically with a period (named *cyclic-period*) and sliding windows in different groups use the vacant memory parallelly.

**Definition 10.**  $\exists Z \subseteq W, \sum_{w \in Z} \text{Min\_}D(w) \leq \min\{T_D(w) \mid w \in Z\}$ .  $\forall w \in Z$ , if  $W_N(w) \geq \text{Min\_}T(w) - \text{Min\_}D(w)$  holds at any time, we call the set  $Z$  as a *cyclic-group* over  $W$ , and call the value of  $\max\{R(w) \mid w \in Z\}$  as the *group-share-memory* of  $Z$ , denoted by  $\text{Share}(Z)$ . Meanwhile, we call the value of  $\min\{T_D(w) \mid w \in Z\}$  as the *cyclic-period* of  $Z$ , denoted by  $P_T(Z)$ .

In a word, when  $M < \sum_{w \in W} \text{Min\_}T(w) \cdot \text{Size}(w) \cdot \text{Rate}(w)$  holds, we should adopt an optimal strategy to group all sliding windows into cyclic-groups and minimize the total size of memory occupied by sliding windows. Supposing  $C$  denotes the set of cyclic-groups, which is formed by the optimal strategy, then the total buffer size of sliding windows is  $M' = \sum_{S \in C} \text{Share}(S) + \sum_{w \in W} \text{Static}(w)$ . Thus, the problem of adjusting the window-width can be described as follows.

**Input:**  $W$  and  $Q$ .

**Output:** The set  $C$  of cyclic-groups over  $W$  with  $C \subseteq 2^W$ .

**Object function:**  $\min \sum_{S \in C} \text{Share}(S) + \sum_{w \in W} \text{Static}(w)$ , i.e.  $\min \sum_{S \in C} \text{Share}(S)$ ,  
where  $\text{Share}(S) = \max\{R(w) \mid w \in S\}$ .

**Subject to:** (1)  $\forall S \in C, \sum_{w \in S} \text{Min\_}D(w) \leq \min\{T_D(w) \mid w \in S\}$ , that is,  $S$  must be a cyclic-group.

(2)  $\bigcup_{S \in C} S = W$ .

(3)  $\forall U \in C, \forall V \in C$ , if  $U \neq V$ , then  $U \cap V = \emptyset$ . (3-2)

**Theorem 2.** When  $M < \sum_{w \in W} \text{Min\_}T(w) \cdot \text{Size}(w) \cdot \text{Rate}(w)$  holds, the problem of adjusting window-width is NP-hard.

Since the time and space complexity of the problem (3-2) with the enumeration method both are  $O(4^d)$ , we propose an optimal algorithm with exponential complexity and an approximation algorithm with polynomial complexity.

Following presents the optimal exponential algorithm, named Optimal-Dynamic-Delay-Based (ODDB). The main idea of ODDB is to number the sliding win-

dows as natural numbers firstly, that is,  $\forall w_i \in W$ , let  $f(w_i)=i$  with  $1 \leq i \leq d$ . Then we map any subset of  $W$  into a natural number between 1 and  $2^d$  using the mapping function  $F$ , that is,  $\forall A \subseteq W$ , let  $F(A)=\sum_{w_i \in A} 2^{f(w_i)}=n$  and denote  $A$  as  $F^{-1}(n)$  with  $1 \leq n \leq 2^d$ . So, any integer between 1 and  $2^d$  corresponds to a set of sliding windows uniquely. Let the variable  $Is\_Group[n]$  determine whether the set  $F^{-1}(n)$  of sliding windows is a cyclic-group, the variable  $Cost[n]$  preserve the optimal cost of the set  $F^{-1}(n)$  and the variable  $split\_point[n]$  be the optimal solution of the set  $F^{-1}(n)$ . Based on above notations, we give the optimal algorithm of ODDB using dynamic programming method.

**Input:**  $W$  and  $Q$ .

**Output:** The set  $C$  of cyclic-groups over  $W$  with  $C \subseteq 2^W$ .

**Optimal-Dynamic-Delay-Based ( )**

- (1)  $\forall w_i \in W$ , let  $f(w_i)=i$  with  $1 \leq i \leq d$ ;
- (2) **for** ( $i=0$ ;  $i < d$ ;  $i++$ )
- (3)  $n=2^i$ ; /\*Producing a cyclic-group  $F^{-1}(n)$  which only contains  $w_i$ \*/
- (4)  $Is\_Group[n]=true$ ;  $Cost[n]=R[i]$ ; /\*Denoting  $F^{-1}(n)$  as a cyclic-group \*/
- (5) **for** ( $j=1$ ;  $j < n$ ;  $j++$ )
- (6)  $m=n$  OR  $j$ ; /\*Unioning  $\{w_i\}$  and  $F^{-1}(j)$ , and producing a new set  $F^{-1}(m)$  \*/
- (7) **if** ( the set  $F^{-1}(m)$  is a cyclic-group )
- (8)  $Is\_Group[m]=true$ ;  $Cost[m]=\max\{Cost[j], R[i]\}$  ;
- (9) **else**
- (10)  $Is\_Group[m]=false$ ; /\*Identifying the set  $F^{-1}(m)$  as a none cyclic-group\*/
- (11)  $Cost[m]=\min\{Cost[k]+Cost[m \text{ XOR } k] \mid 1 \leq k < j \wedge F^{-1}(k) \subseteq F^{-1}(j)\}$

The space complexity of ODDB is  $O(2^d)$  and the time complexity of ODDB algorithm is  $O(\sum_{i=0}^{d-1} \sum_{j=1}^{2^i-1} (j-1)) = O(\frac{1}{2} \sum_{i=0}^{d-1} (2^{2i} - 3 \cdot 2^i + 2)) = O(\frac{1}{6} 4^d - \frac{3}{2} 2^d + d + \frac{4}{3}) = O(4^d)$ .

**Theorem 3.** When  $M < \sum_{w \in W} Min\_T(w) \cdot Size(w) \cdot Rate(w)$  holds, ODDB algorithm can produce the optimal solution.

Following is the polynomial approximate algorithm, named Approximate-Delay-Based (ADB). The main idea of ADB algorithm is to cluster the sliding windows into a cyclic-group whose delay-memory is comparative in order to minimize the sum of group-share-memory.

**Input:**  $W$  and  $Q$ .

**Output:** The set  $C$  of cyclic-groups over  $W$ .

**Approximate-Delay-Based ( )**

- (1) Ranking all sliding windows descendingly in terms of the delay-memory and form the queue  $w_1, \dots, w_p, \dots, w_d$ ;
- (2)  $k=1$ ;  $n=0$ ; /\*  $k$  denotes the suffix of the sliding window, and  $n$  denotes the number of cyclic-groups. \*/
- (3) **while** ( $k \leq d$ )
- (4)  $l=1$ ; /\*  $l$  denotes the suffix of the cyclic-group.\*/
- (5) **while** ( $l \leq n$ )
- (6) **if** ( $\{w_k\} \cup Z_l$  is a cyclic-group)  $Z_l = \{w_k\} \cup Z_l$ ; **Break**;
- (7) **else**  $l++$ ;
- (8) **if** ( $l > n$ )  $Z_l = \{w_k\}$ ;  $n++$ ; /\* let  $w_k$  form a new cyclic-group by itself\*/
- (9)  $k++$ ;
- (10) **Output** the set  $C = \{Z_l \mid 1 \leq l \leq n\}$ ;

The time complexity of ADB algorithm is  $O(d^2)$  and the space complexity is  $O(d)$ .

## 4 Experimental Results

To evaluate the efficiency of our proposed algorithms, we perform the simulative experiment with random data. During each experiment, we randomly produce several sliding windows and 600 continuous queries and adjust the window-width of sliding windows using the corresponding algorithms in section 3. Every experimental result is the average of 10 times.

The content of the experiment is divided into three parts. The first part is to examine the performance of algorithms. The second part is to study the error of continuous queries influenced by the adjustment mechanism. The third part is to examine the availability of the memory influenced by the adjustment mechanism. All parts of the experiment will show that our work is significant to the data stream system.

As the experiment for the performance of A-rank-adjusting algorithm and B-rank-adjusting algorithm is simple, we omit it. So, we mainly examine the performance of C-rank-adjusting algorithm and study the difference of performance between ODDb algorithm and ADB algorithm in the environment of data steam system with the variable number of sliding windows. Since the time and space complexity of ODDb algorithm is very high, we can only get the optimal solution of the small-scale problem. So the number of the sliding windows is limited below 17. Figure 1 presents the difference of performance between ODDb algorithm and ADB algorithm as the number of sliding windows changes. At any time, the difference of performance is not great and the relative difference does not exceed 20%. It implies that the approximate ADB algorithm can perform wonderfully for the large-scale problem.

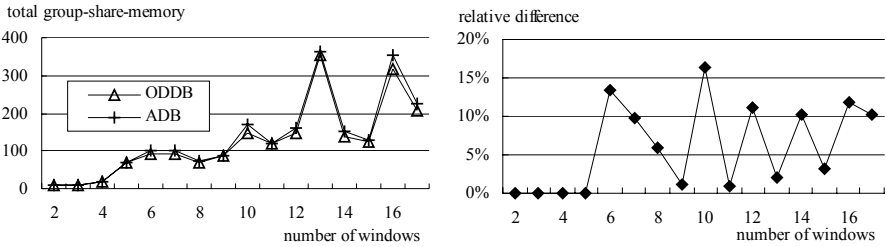


Fig. 1. ODDb vs. ADB

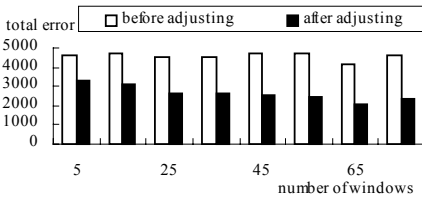


Fig. 2. Difference of total error

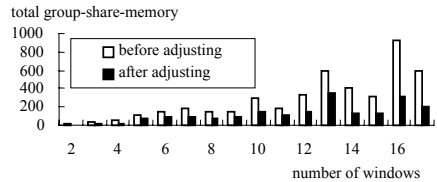


Fig. 3. Difference of total group-share-memory

The experiment result about impact of the adjustment mechanism upon the total error of queries is shown in Figure 2. We examine the difference of the sum of window-accumulate-error before and after using the adjustment mechanism in the envi-

ronment of data steam system with the variable number of sliding windows. As shown in Figure 2, the total the window-accumulate-error changes a lot before and after adjusting the window-width. The experiment result shows that our adjustment mechanism decreases a lot of the total error of continuous queries.

Figure 3 shows the impact of the adjustment mechanism upon the availability of memory. We study the difference of the sum of group-share-memory before and after using the adjustment mechanism in the environment of data steam system with the variable number of sliding windows.

Before using the adjustment mechanism, there is no sharing of the memory among the sliding windows, that is, the number of sliding windows in each cyclic-group is 1. However, after using the adjustment mechanism, more than one sliding window in each cyclic-group can share the memory, so that low memory overhead is achieved. As shown in Figure 3, the total group-share-memory changes a lot before and after adjusting the window-width, which shows that our adjustment mechanism saves a lot of the memory, that is, it achieves low memory overhead.

## 5 Summary

In this paper, we propose the algorithms of adjusting the buffer size of sliding windows in the data stream system with limited memory, aiming to minimize the total error of continuous queries or achieve low memory overhead. Analytical and experimental results show that our algorithms can be applied to the data stream system efficiently.

## References

1. S. Guha, N. Koudas. Approximating a Data Stream for Querying and Estimation: Algorithms and Performance Evaluation. ICDE, 2002.
2. L. Golab, M. T. Ozsü. Processing Sliding Window Multi-Joins in Continuous Queries over Data Streams. VLDB, 2003.
3. S. D. Viglas, J. F. Naughton, J. Burger. Maximizing the Output Rate of Multi-Way Join Queries over Streaming Information Sources. VLDB, 2003.
4. M. A. Hammad, M. J. Franklin, W.G. Aref and A.K. Elmagarmid. Scheduling for shared window joins over data streams. VLDB, 2003.
5. B. Babcock, S. Babu, M. Datar, and R. Motwani. Chain: Operator Scheduling for Memory Minimization in Data Stream Systems. SIGMOD, 2003.
6. P. J. Haas. Large-Sample and Deterministic Confidence Intervals for Online Aggregation. 9th International Conference on Scientific and Statistical Database Management (SSDBM, 1997).
7. J. M. Hellerstein, P. J. Haas, H. J. Wang. Online Aggregation. SIGMOD, 1997.
8. A. Arasu, B. Babcock and S.Babu. Characterizing Memory Requirements for Queries over Continuous Data Streams. PODS, 2002.
9. J. Kang, J. F. Naughton, S. D. Viglas. Evaluating Window Joins over Unbounded Streams. ICDE, 2003.
10. A. Araru, S. Babu, J. Widom. An Abstract Semantics and Concrete Language for Continuous Queries over Streams and Relations. Technical Report, Stanford University Database Group. Nov. 2002. Available at <http://dbpubs.stanford.edu/pub/2002-57>.

# Space Efficient Quantile Summary for Constrained Sliding Windows on a Data Stream

Jian Xu<sup>1</sup>, Xuemin Lin<sup>1</sup>, and Xiaofang Zhou<sup>2</sup>

<sup>1</sup> Computer Science & Engineering  
The University of New South Wales  
Sydney 2052, Australia  
{xujian, lxue}@cse.unsw.edu.au

<sup>2</sup> School of ITEE  
The University of Queensland  
Brisbane QLD 4072, Australia  
zxf@itee.uq.edu.au

**Abstract.** In many online applications, we need to maintain quantile statistics for a sliding window on a data stream. The sliding windows in natural form are defined as the most recent  $N$  data items. In this paper, we study the problem of estimating quantiles over other types of sliding windows. We present a uniform framework to process quantile queries for time constrained and filter based sliding windows. Our algorithm makes one pass on the data stream and maintains an  $\epsilon$ -approximate summary. It uses  $O(\frac{1}{\epsilon^2} \log^2 \epsilon N)$  space where  $N$  is the number of data items in the window. We extend this framework to further process generalized constrained sliding window queries and proved that our technique is applicable for flexible window settings. Our performance study indicates that the space required in practice is much less than the given theoretical bound and the algorithm supports high speed data streams.

## 1 Introduction

In many applications, massive volume of data is generated and collected from distributed sources and needs to be processed in real time. The data can be modelled as an unbounded data sequence arriving at a port of the system thus is given the name “**data stream**”. The uprising requests of processing data streams call for a revolutionary view and new techniques. In stream processing, a data item can be accessed at its arriving time only once. As the size of a data stream is often unbounded, processing needs to make as small size memory footprints as possible. On one hand, one pass of strict ordered scan of data makes save both on access time and storage. On the other hand, many operations such as join and aggregations become non-trivial under data stream model. Elegant algorithms are needed to meet the space and time restrictions.

New semantic for processing data streams keeps being discovered. Among those, processing data streams over sliding windows receives much research attention[1, 2]. As data arrives in order, it is natural to define “old” data which is

observed earlier and “recent” ones which has just arrived. In many applications, only recent data items are interested to the user. Therefore, computations need to focus on a sliding window which contains only a portion of most recent data in the stream. In this paper, we study quantile problems on sliding windows.

Quantile information unveils important information on the data distribution of a data set. The task is to compute  $\phi$  quantile, which is the value of the data at rank  $\lceil \phi N \rceil$  (after sorting) for a set of  $N$  data elements. Early in 1980, Munro and Paterson proved in [5] an  $\Omega(N^{\frac{1}{p}})$  lower bound on space requirement for computing exact quantiles over  $N$  data elements in  $p$  passes of scanning the data. In data stream, as it’s impossible to get accurate quantiles, it is desirable to have estimations with accuracy guarantees. The work [7, 8] studied the problem of estimating approximate quantiles for whole data stream. The space requirement is  $O(\frac{1}{\epsilon} \log^2 \epsilon N)$  for  $\epsilon$ -approximate quantiles. In a recent paper [4], this problem was re-studied for append only data stream. A deterministic algorithm using a space of  $O(\frac{1}{\epsilon} \log \epsilon N)$  was presented. In [3], the authors studied the problem on a data stream that has both append and deletion. The algorithm uses  $O(\log^2 |U| \log(\log(|U|)/\delta)/\epsilon^2)$  space, given the value domain  $U$  of data and the algorithm estimates  $\epsilon$ -approximate quantiles with confidence  $1 - \delta$ . The only work we know for estimating quantiles on sliding windows is [6]. In this paper, the authors gave a solution for estimating quantiles on a sliding window containing a fixed number of data items. The algorithm uses  $O(\frac{1}{\epsilon^2} + \frac{1}{\epsilon} \log \epsilon^2 N)$  space for sliding window containing  $N$  data items. In addition to the natural form of sliding window, a window may be constrained to other parameters such as “*data arrived in most recent 5 minutes*” or “*phone calls originated from CSE in the most recent 10000 calls inside campus*”. Challenges on such sliding windows lie on two aspects. Firstly, the number of data items in the window is unknown and changes over time. The technique in [6] does not work without an apriori to the window size. Secondly, constraints may have various of forms and a practical query processing should be able to handle different window queries in a uniform way rather than hacking from one specific solution to another. Our contribution can be summarized as following:

1. We give a sub-linear deterministic algorithm for estimating  $\epsilon$ -approximate quantiles over time constrained and filter based sliding windows.
2. We give a uniform framework for estimating  $\epsilon$ -approximate quantiles under an extended sliding window definition, using sub-linear space.

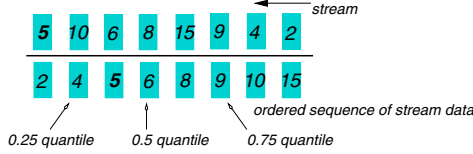
## 2 Preliminary

In this section, we introduce some background of quantile estimation over data streams and define terms to use in the later sections.

### 2.1 Quantile Computations on Data Stream

**Definition 1 (Quantile).** A  $\phi$ -quantile ( $\phi \in (0, 1]$ ) of an ordered sequence of  $N$  data items is the data item at rank  $\lceil \phi N \rceil$ .





**Fig. 1.** Quantiles of a sequence

Figure 1 is an example for quantiles on a data stream of 8 items. As the difference between  $\phi N$  and  $\lceil \phi N \rceil$  is smaller than 1, without loss of generality, we use  $\phi N$  as the requested quantile point instead of  $\lceil \phi N \rceil$  in the paper. Definition 2 gives the quantitative approximation measurement.

**Definition 2 ( $\epsilon$ -Approximate Quantile).** An  $\epsilon$ -approximate answer for a quantile query  $Q(\phi)$  is a data item of which the rank  $r$  satisfies  $|r - \phi N| \leq \epsilon N$ . Where  $N$  is the number of data elements in the sequence.

Generally, a summary which estimates  $\epsilon$ -approximate quantiles may be in any form. In our study, we use *quantile sketch*, or *sketch* for brevity, to denote a summary structure proposed in [4], as the quantile summary of a data stream.

**Definition 3 (Quantile Sketch).** A quantile sketch  $S$  of an ordered data sequence  $D$  is an ordered sequence of  $m$  tuples  $\{(v_i, r_i^-, r_i^+) : 1 \leq i \leq m\}$  with the following properties.

1. Each  $v_i \in D$ .
2.  $v_i \leq v_{i+1}$  for  $1 \leq i \leq m - 1$ .
3.  $r_i^- < r_{i+1}^-$  for  $1 \leq i \leq m - 1$ .
4. For  $1 \leq i \leq m$ ,  $r_i^- \leq r_i \leq r_i^+$  where  $r_i$  is the rank of  $v_i$  in  $D$ .

An  $\epsilon$ -approximate sketch is also written as  $\epsilon$ -sketch for short. And Greenwald and Khanna proved the following lemma in [4].

**Lemma 1 (GK  $\epsilon$ -sketch).** An  $\epsilon$ -sketch satisfies the following conditions.

1. The first tuple  $r_1^+ \leq \epsilon N + 1$ ,
2. The last tuple  $r_m^- \geq (1 - \epsilon)N$ ,
3. for  $2 \leq i \leq m$ ,  $r_i^+ \leq r_{i-1}^- + 2\epsilon N$ .

To query the sketch for quantile  $\phi$ , the algorithm scans the sketch and returns a tuple  $(v, r^-, r^+)$  which  $r^- > \phi N - \epsilon N$  and  $r^+ < \phi N + \epsilon N$ . It was shown in [4] that maintaining such a sketch requires  $O(\frac{1}{\epsilon} \log \epsilon N)$  space. We use the term **GK-sketch** to refer to an instance of a sketch maintained by algorithm in [4].

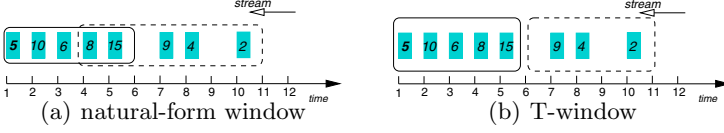


Fig. 2. Two types of sliding windows

## 2.2 Constrained Sliding Windows

The sliding window studied in [6] can be viewed as of a natural-form. It is defined on the number of data items in the window. For the windows defined on other constraints, we call them “*constrained sliding windows*”. Fig. 2 shows the difference between the two types of windows. A strict definition on accepted constraints to a sliding window will be derived in section 4, after we show the framework for estimating quantiles for two typical constrained windows.

## 3 The Sliding Window Techniques

In this section, we present a framework for estimating approximate quantiles over constrained sliding windows and show how quantiles can be estimated for *time constrained* and *filter based* sliding windows.

### 3.1 The Framework

The basic idea of our algorithm is to partition the stream along its incoming order and build sketches on each partition so that at any time, a query can be answered by extracting an appropriate sketch from the summary.

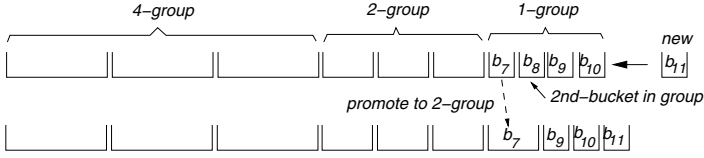
The validity of partitioning a data stream while still keeping the accuracy of quantile estimations is based on the following theorem.

**Theorem 1.** *Given an  $\frac{\epsilon}{2}$ -sketch  $S_n$  on the most recent  $n$  data items, then a  $\phi$ -quantile query estimation from  $S_n$  is an  $\epsilon$ -approximate  $\phi$ -quantile for the most recent  $N$  data items, if  $n \leq N \leq (1 + \frac{\epsilon}{2})n$ .*

To build a summary, we partition the stream into  $k$  buckets  $b_1, b_2, \dots, b_k$ . For each bucket, we maintain an  $\frac{\epsilon}{2}$ -sketch which covers all data items that arrived after the bucket was constructed. Let the number of data covered by the sketch in  $b_i$  be  $n_i$ . The buckets in the summary satisfy that  $n_i - n_{i+1} \leq \frac{\epsilon}{2}n_{i+1} + 1$  and  $n_i - n_{i+2} > \frac{\epsilon}{2}n_{i+2} + 1$ , for  $1 \leq i < k - 1$ , namely  $\frac{\epsilon}{2}$ -partition.

We build in each bucket an  $\frac{\epsilon}{2}$ -approximate GK-sketch and according to theorem 1, a sketch can be extracted from the summary for any length of window. We use  $(a, b)$ -summary to denote a summary built on  $a$ -partition and  $b$ -sketch. e.g. The summary we just built is an  $(\frac{\epsilon}{2}, \frac{\epsilon}{2})$ -summary.

**Lemma 2.** *The number of buckets maintained in an  $\epsilon$ -partitioned summary, if  $n_1 = N$ , is  $\Theta(\frac{1}{\epsilon} \log \epsilon N)$ .*



**Fig. 3.** Drop redundant buckets in summary

As  $O(\frac{1}{\epsilon} \log \epsilon N)$  space is needed for a GK-sketch. We have,

**Theorem 2.** *The total space needed by an  $(\epsilon, \epsilon)$ -summary is  $O(\frac{1}{\epsilon^2} \log^2 \epsilon N)$ .*

Let  $d_i = n_i - n_{i+1}$  and we group buckets with same  $d_i$  to form “ $d$ -groups”. (see Fig. 3) Observe that dropping of buckets will always occur on the second bucket in an  $d$ -group, promoting the first one into the  $2d$ -group. The  $d_i$  thus takes a form of  $2^j$ ,  $j = 0, 1, 2, \dots$ . Use the space bound we derived, there will be  $\log \epsilon N$  such  $d$ -groups and each contains no more than  $\lfloor \frac{1}{\epsilon} \rfloor + 1$  buckets. We give algorithm 1 to efficiently maintain the summary structure.

---

**Algorithm 1** Summary Maintenance

---

Upon a new data item  $v$  arrives:

**step 1:** Allocate a new bucket for this data item and append it to 1-group.

**step 2:** Iteratively checking the  $d$ -groups, from 1-group upwards. If the  $i$ -group is full (i.e. contains  $\lfloor \frac{1}{\epsilon} \rfloor + 2$  buckets), drop the second bucket in this group and move the first one to  $2i$ -group. Stop iteration when at  $i_0$ -group, no bucket is dropped.

**step 3:** Add  $v$  to sketches maintained in all the buckets in the summary.

---

It is important to note here that although in the analysis of the algorithm, we use  $N$ ,  $n$  to refer to the number of data items in the window, it does not have to be unveiled to the algorithm. Therefore, the sliding window does not have to define on “item counting”. Additionally, for any window length, we can find a corresponding sketch in the summary. This enables answering queries for different window lengths using one single summary.

### 3.2 Quantile Summary for Two Constrained Sliding Windows

**T-Window Quantile Query:** A T-window query with parameters  $\phi$ ,  $T$  and  $\epsilon$  can be written in SQL-like syntax as:

SELECT  $\phi$ -quantile FROM stream

RANGE= data arrived in last  $T$  time

PRECISION =  $\epsilon_0$

Here, the parameter  $\phi$  and  $T$  can vary from query to query while  $\epsilon_0$  is pre-fixed when the summary is built.

We build an  $(\frac{\epsilon}{2}, \frac{\epsilon}{2})$ -summary and record the time when each bucket is built. To query a quantile, we choose the earliest constructed bucket with time-stamp inside the query window and estimate quantile using the sketch in this bucket.

**N,f-Window Quantile Query:** An N,f-window quantile query with parameters  $\phi$ ,  $N$ ,  $\epsilon$  and filter function  $f$  can be written in SQL-like syntax as:

SELECT  $\phi$ -quantile FROM stream WHERE  $f(v) = true$

RANGE= most recent  $N$  data items.

PRECISION =  $\epsilon_0$

The underlined parameters vary from query to query while  $f$  and  $\epsilon_0$  are fixed when building the summary structure.

A single GK-sketch can handle filter based quantile query on whole stream by adding only the data items that pass the filter into the sketch. While according to [6], an approximate quantile summary for an N-item window can be maintained. However, neither can handle this  $N, f$ -window query.

To process the  $N, f$ -window quantile query, we build an  $(\frac{\epsilon}{2}, \frac{\epsilon}{2})$ -summary. A data item  $v$  is added into the summary if  $f(v) = true$ . On each bucket  $b_i$ , we record the number  $n_i$  of data items observed (whatever added or not to the summary) after this bucket is built. To query a quantile, we scan the summary and choose the first bucket  $b_k$  that  $n_k \leq N$ , and return the quantile estimated by the sketch in it.

## 4 Processing General Queries

In this section, we propose a uniform summarization technique for processing quantile queries on sliding windows with general constraints.

### 4.1 Sliding Window and Regression Function

A sliding window has the following properties.

1. **Monotony.** An expired data item won't re-enter a window.
2. **FIFO.** Early arrived data items expires from a window before late items.

We first define *monotonic boolean function*, as following.

**Definition 4 (Monotonic Boolean Function).** A monotonic boolean function  $F : V \mapsto \{true, false\}$  satisfies,

$$\exists t_0, F(v)|_{t_0} = false \longrightarrow \forall t > t_0, F(v)|_t = false.$$

Here,  $t > t_0$  means a time  $t$  later than  $t_0$  and  $V$  is the domain of the attribute( $s$ ) tested by  $F$ .

We define *regression function*(RF) to fully specify a sliding window.

**Definition 5 (Regression Function (RF)).** A function  $R$  is a regression function if it satisfies the following two conditions:

1.  $R$  is a monotonic boolean function.
2. For two data items  $v_1$  and  $v_2$  which  $v_1$  arrives earlier than  $v_2$ , if  $R(v_1)$  and  $R(v_2)$  are both true at time  $t_0$ , then there is no time  $t > t_0$  that  $R(v_1) = true$  while  $R(v_2) = false$ .

We use the term *R-window*, or simply *R* to refer to a sliding window constrained by *R*. And the sliding windows studied in the last section is described using regression function like this:

1.  $R_N(v) = \text{true}$  iff  $v$  is among the most recent  $N$  items.
2.  $R_T(v) = \text{true}$  iff  $v$  is observed in recent  $T$  time.
3.  $R_f(v) = f(v)$ ,  $f(v) : V \mapsto \{\text{false}, \text{true}\}$

We specially address on  $R_f$  that, for a given function  $f$ , a data item is determined to be inside( $f(v) = \text{true}$ ) or outside( $f(v) = \text{false}$ ) of the window at the time it is observed and the status never changes afterwards. We define *filter RF* specially for this kind of regression function.

**Definition 6 (Filter RF).** *Function  $R$  is a filter RF if*

1.  $R$  is a regression function,
2. If at time  $t_0$ ,  $\exists v, R(v) = \text{true}$ , then  $R(v) = \text{true}$  for all  $t > t_0$ .

It is easy to see that composition of two RFs under **AND** or **OR** is still a RF. However, only filter-RFs keep this property under **NEG** operation.

We are now ready to show how a query on a generally constrained window, possibly a boolean expression formed by regression functions, be processed in a uniform manner in our framework.

## 4.2 Processing Quantile Queries for a General *R*-Window

A constraint  $R$  can be written in DNF.

$$\begin{aligned}
 R = & (a_{11} \wedge a_{12} \dots \wedge a_{1m_1} \wedge b_{11} \wedge b_{12} \dots \wedge b_{1n_1}) \\
 & \vee (a_{21} \wedge a_{22} \dots \wedge a_{2m_2} \wedge b_{21} \wedge b_{22} \dots \wedge b_{2n_2}) \\
 & \dots \\
 & \vee (a_{j1} \wedge a_{j2} \dots \wedge a_{jm_j} \wedge b_{j1} \wedge b_{j2} \dots \wedge b_{jn_j})
 \end{aligned}$$

Here, regression functions are divided into classes  $a$  and  $b$ , where class  $b$  is the class of filter-RFs and  $a$  for non-filter RFs. We use  $t_i$ ,  $1 \leq i \leq j$  to denote the  $j$  conjunction clauses. We build the summary in the following steps.

**Step 1: Scan for pure class  $a$  clause** If there exists one or more  $n_i = 0$  ( $1 \leq i \leq j$ ). We form a “pure class  $a$  sub-constraint”  $A = \bigvee_{\{i:n_i=0\}} t_i$ .

**Step 2: Scan for pure class  $b$  clause** If there exists one or more  $m_i = 0$  ( $1 \leq i \leq j$ ). We form a “pure class  $b$  sub-constraint”  $B = \bigvee_{\{i:m_i=0\}} t_i$ .

The remaining clauses form constraint  $C = \bigvee_{\{i:m_i*n_i \neq 0\}} t_i$  so  $R = A \vee B \vee C$ .

**Step 3: Build quantile summary** Algorithm 2 demonstrates this procedure.

When a new data item arrives in the data stream, the summary is updated using algorithm 3. It is easy to verify that the space usage of the whole summary is bounded by  $O(\frac{1}{\epsilon^2} \log^2 \epsilon n_1 + \frac{1}{\epsilon} \log \epsilon n_2)$ , where  $n_1, n_2$  is the number of items inserted into  $S1, S2$  respectively. To query the summary for  $\phi$ -quantile on an *R*-window, first extract sketch  $sk_R$  from the  $(\frac{\epsilon}{2}, \frac{\epsilon}{2})$ -summary  $S1$  for the query

**Algorithm 2 Build Summary****Input:**Constraints  $R = A \vee B \vee C$ .**Output:**Quantile Summary for  $R$ -window.**Description:**

- 1: **if**  $B$  is not an empty constraint **then**
- 2:   Build a GK-sketch  $S2$  on error parameter  $\epsilon$ ;
- 3: **end if**
- 4: **if**  $C$  is not an empty constraint **then**
- 5:   Form constraint  $B_s = (\bigvee_{\{i:t_i \text{ in } C\}} t_i^b) \wedge \neg B$ ; ( $t_i^b = \bigvee_{k=1}^{n_i} b_{ik}$ )
- 6: **end if**
- 7: Build an  $(\frac{\epsilon}{2}, \frac{\epsilon}{2})$ -summary  $S1$  with constraints  $B_s$  and  $A \vee C$ ;
- 8: **return**  $S = (S1, S2)$ ;

**Algorithm 3 Summary Maintenance****Input:**Summary  $S(S1, S2)$  built in algorithm 2 and a data item  $v$ .**Output:**

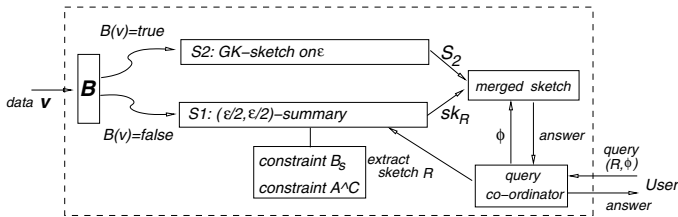
Updated summary.

**Description:**

- if**  $B(v) == \text{true}$  **then**
- Add  $v$  into sketch  $S2$ .
- else if**  $B_s(v) == \text{true}$  **then**
- Add  $v$  into summary  $S1$ . (use algorithm 1)
- end if**

window. Then *merge*  $sk_R$  with  $S2$ . The merged sketch is an  $\epsilon$ -sketch and we query it for  $\phi$ -quantile.

We omit the detail of operation *merge* here due to space limitation. A general version (of merging multiple sketches) can be found in [6] in which it is proved that the merged sketch is  $\epsilon$ -approximate. The whole diagram of maintenance and query of the summary is shown in figure 4.

**Fig. 4.** Working flow of the framework

## 5 Performance Study

We present performance evaluations as well as analysis on practical issues of our algorithm in this section. The algorithms are implemented in C++ and compiled using GNU gcc(v2.95). We run experiments on a P4 2.0GHz , 512MB memory PC with Debian Linux as its operating system.

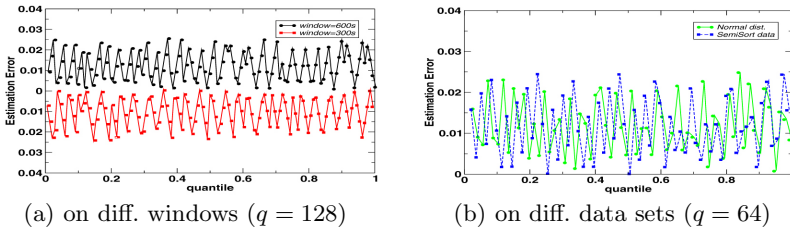
### 5.1 Performance Evaluation on Synthetic Data Sets

We test the algorithm using a set of T-window queries. The parameters that affect experiments are listed in table 1 with their default values used in this subsection.

**Table 1.** T-window query: system parameters

Notation	Definition (Default Values)
$\epsilon$	The guaranteed precision (0.05)
$\phi$	$(1/q, \dots, (q-1)/q)$ for a given $q$ ( $q=64$ )
$D$	Data distribution of the stream (uniform)
$Sr$	Stream Rate (approx. 400 tuples/s)
$Mw$	Max window length(on time) (600s)
$Ss$	Stream Size ( $1.5 * Mw * Sr = 360K$ )
$Qw$	Query window length(on time) (600s)

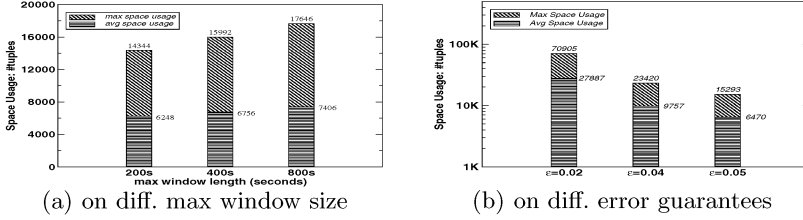
Figure 5(a) shows the estimation errors for quantiles from  $1/128..127/128$  on  $Qw = 600s$  and  $Qw = 300s$ . Figure.5(b) shows the experiment results using normal distributed data and semisort data. From the result we can see that under a guaranteed error setting of 0.05, the estimation errors are much lower than the guaranteed value. We also have similar observations on summaries with other  $\epsilon$ , omitted here due to space limitation.



**Fig. 5.** Estimating accuracy evaluation

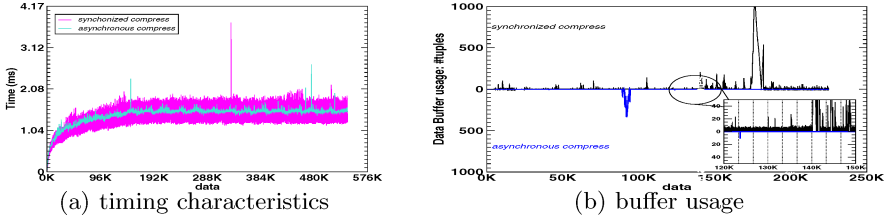
We also measure the memory space used by the summary. Figure 6 shows the space consumption under different settings. We present two statistics in the result graphs. The “max space usage” is the peak value of memory consumption and “avg space usage” represents the average space consumption. Figure 6(a) shows the space consumption when  $Mw$  varies from 200s, 400s, to 800s. We can

see that the space consumption increases very slowly on window enlargement, both for max and avg space usage. In Fig. 6(b), we compare the space usage when precision requirement changes. The experiment results show that the space usage increases apparently when  $\epsilon$  gets smaller and the trend follows our analysis of the space usage.



**Fig. 6.** Performance of space consumption: T-window

We conducted a set of experiments to measure the timing characteristics of the algorithm. Compressing of the GK-sketch dominates the running time of the algorithm. As we maintain multiple sketches in the summary, avoiding a synchronized compression of all sketches can have benefits that both the overall runtime performance is less fluctuating and the use of data buffer for incoming stream items can be reduced.



**Fig. 7.** Runtime performance: synchronized v.s. asynchronous

We tune our algorithm to compress sketches asynchronously. The result is shown in Fig. 7 where in sub-figure (a), the fluctuation of asynchronous compress (pale green curve inside) is much less than that of synchronized compression. Time needed for an individual insertion is calculated by an average of a batch of 16 consecutive insertions to reduce measuring error. Figure 7(a) also suggests that the algorithm supports high speed data stream ( $> 500$  tuples/s). Figure 7(b) shows the buffer usage of the algorithm, the stream rate is 500 tuples per second. The difference on buffer usage between synchronized and asynchronous compressions is apparent from the experiment that asynchronous compression needs generally less buffer in most of the time.



## 5.2 Real Data Set Evaluations

We also test the algorithm using a real data set. We use data from TDT(Topic Detection and Tracking) data repository. The data collected from news articles and stories is hashed to numerical values for the experiments.

Figure8 shows the accuracy and space consumption tests on default settings using the real data set for T-window queries(Fig.8(a)). We also test a set of N,f-window queries, in which the selectivity of filter function is set to 0.6 and window lengths are 100K and 200K respectively. The results are shown in Fig.8(b).

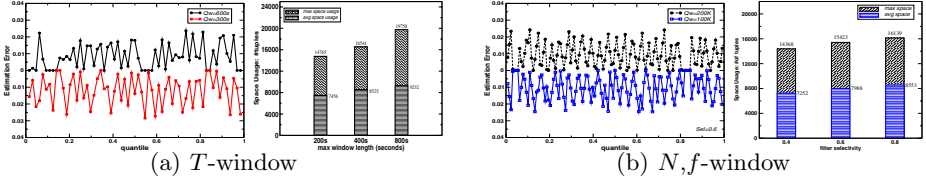


Fig. 8. Real data experiments

## 6 Conclusion

In this paper, we study the problem of processing approximate quantile queries for constrained sliding windows on a data stream. We present a framework that estimates  $\epsilon$ -approximate quantiles using  $O(\frac{1}{\epsilon^2} \log^2 \epsilon N)$  space. We extend the framework to process generally constrained sliding window queries, using  $O(\frac{1}{\epsilon^2} \log^2 \epsilon n_1 + \frac{1}{\epsilon} \log \epsilon n_2)$  space ( $n_1 + n_2 = N$ ).

## References

1. B. Babcock, M. Datar, R. Motwani, and L. O’Callaghan. Sliding window computations over data streams. In *ACM PODS*, 2003.
2. M. Datar, A. Gionis, P. Indyk, and R. Motwani. Maintaining stream statistics over sliding windows. In *13th ACM-SIAM symposium on Discrete algorithms*, 2002.
3. A. C. Gilbert, Y. Kotidis, S. Muthukrishnan, and M. J. Strauss. How to summarize the universe: Dynamic maintenance of quantiles. In *VLDB*, 2002.
4. M. Greenwald and S. Khanna. Space-efficient online computation of quantile summaries. In *ACM SIGMOD*, 2001.
5. J.I.Munro and M.S.Paterson. Selection and sorting wiith limited storage. In *TCS12*, 1980.
6. X. Lin, H. Lu, J. Xu, and J. X. Yu. Continuously maintaining quantile summaries of the most recent n elements over a data stream. In *ICDE*, 2004.
7. G. S. Manku, S. Rajagopalan, and B. G. Lindsay. Approximate medians and other quantiles in one pass and with limited memory. In *ACM SIGMOD*, 1998.
8. G. S. Manku, S. Rajagopalan, and B. G. Lindsay. Random sampling techniques for space efficient online computation of order statistics of large datasets. In *ACM SIGMOD*, 1999.

# The Dimension-Reduced Multi-scale Histogram: A New Method for Fast and Efficient Retrieval of Similar Time Sequences and Subsequence

Jian-Wei Liu, Shou-Jian Yu, and Jia-Jin Le

College of Information Science and Technology, Donghua University  
Shanghai, 200051, China  
{liujw, jackyyys}@mail.dhu.edu.cn  
lejiajin@dhu.edu.cn

**Abstract.** In this paper, we propose dimensionality reduction representation of multi-scale time series histograms, which is performed based on the multi-scale histograms. It is a faster and efficient way to pre-select time sequence in a database and leads to reduce the need of time sequence comparisons when answering similarity queries. A new metric distance function MD ( ) that consistently lower-bounds WED and also satisfies the triangular inequality is also presented and based on it, we construct the Slim-tree index structure as the metric access method to answer similarity queries. We also extend it to subsequence matching and presented a MSST index structure.

## 1 Introduction and Related Works

Time-series data are of growing importance in many new database applications, such as data mining or data warehousing. The problem of retrieving similar time sequences may be stated as follows:

Given a query sequence  $q$ , a set of time sequences  $X = [(x_1, t_1), \dots, (x_n, t_n)]$ , a distance measure  $d$ , and a tolerance threshold  $\epsilon$ , find the set  $R$  of sequences closer to  $q$  than  $\epsilon$ , or more precisely:

$$R = \{x \mid d(q, x) \leq \epsilon\} \quad (1)$$

There are essentially two ways the data might be organized:

**Whole Matching:** Here it assumed that all sequences to be compared are the same length.

**Subsequence Matching:** Here we have a query sequence  $X$ , and a longer sequence  $Y$ . The task is to find the subsequence in  $Y$ , beginning at  $Y_i$ , which best matches  $X$ , and report its offset within  $Y$ .

However, any proposal of such similarity queries technique must be supported by indexing for very large databases. Note that, any indexing scheme that does not examine the entire dataset could potentially suffer from two problems, false alarms and false dismissals. In order to guarantee no false dismissals, the distance in the index space must satisfy the following condition

$$d_{\text{indexspace}}(x, y) \leq d_{\text{true}}(x, y) \quad (2)$$

Generally speaking, efficacy of retrieval methods of similar time sequences depends on used techniques of dimensionality reduction data representation, robust distance measures and good indexing methods. There are many methods for efficient retrieval of similar time sequences: Agrawal et al. in [1] introduced a method called the F-index which a signature extracted from the frequency domain of a sequence; An approach independently introduced by Yi and Faloutsos [2] and Keogh et al. [3] is to divide each sequence into  $k$  segments of equal length, and to use the average value of each segment as a coordinate of a  $k$ -dimensional signature vector; Perng et al. [4] introduces a general landmark model.

Among them, a method of multi-scale histogram-based representation with weighted Euclidean distance to time series data outperform string representations in finding patterns and outperform DTW and LCSS in answering exact match queries when the time series data contain noise or time shifting and scaling [5].

The main contributions of the paper are:

In this work, we present dimensionality reduction representation of multi-scale histograms of time series. The main objective of dimensionality reduction representation of multi-scale histograms of time series was to define a faster way to pre-select time sequence in a database, which is performed based on the multi-scale histograms, to reduce the need of time sequence comparisons when answering similarity queries. The new metric distance function MD ( ) to measure the dissimilarity between time series through their dimension-reduced multi-scale histograms is also presented. We also prove the no false dismissal property of our approach. Finally, we use slim-tree to create hierarchy indexing structures in metric space. Slim-tree index structure minimizes the number of distance calculations taking advantage of the triangular inequality property of metric distance functions. We also extend dimension-reduced multi-scale histograms to subsequence matching. Our techniques extend and generalize the earlier results of Lei Chen et al. [5]. Based on the idea of using multiple resolution levels, we proposed another index structure called MSST-index that clusters the subsequences for different resolutions in a grid of slices. We provide a flexible method that allows the user to specify queries of variable length, and the technique can be tuned to optimize the retrieval time or the accuracy of the solution.

The rest of the paper is organized as follows: In Section 2 point out existing problem and presents our approach and Section 3 contains results obtained from experimentations with real data sets. Finally, in Section 4 we conclude and discuss about possible extensions for our work.

## 2 Existing Problem and Proposed Solutiont

### 2.1 Dimension-Reduced Multi-scale Time Series Histograms Representation of Time Series

Experimental result show that the number of bins is up to 64 bins and scales  $\delta$  is at least equal to 4 in order to achieve reasonable good results in [5]. Sequential scanning

over a large set of time series is impractical due to the high computational cost of comparing two time series. The main practice used to accelerate the data retrieval in time series database is indexing the data through an access method tailored to the domain of the data. Indexing the extracted time series characteristics is usually done through spatial access methods, where each extracted feature is a dimension. Regarding time series histograms, each bin is indexed as a dimension. Indexing histograms like this one requires indexing arrays with 16-256 elements or, in terms of indexing structures, dimensions.

However, the spatial access methods are efficient up to a limited number of dimensions, in general not larger than 10 or 20. Indexing time series through their histograms with several hundred bins or more is highly inefficient. Many attempts have been done to find more concise representations of histograms using dimensionality reduction techniques [6]. All these attempts lead to histograms representations with a reduced number of dimensions.

Given the multi-scale representation of time series histogram, we propose to represent a dimension-reduced multi-scale representation of time series histogram through a set of line segments. In order to construct dimensionality reduction representation of multi-scale histograms of time sequence, we will utilize the Piecewise Aggregate Approximation (PAA) [7]. Specifically, the multi-scale histograms  $H^\delta = [\langle b_1, h_1 \rangle, \dots, \langle b_\tau, h_\tau \rangle]$  of length  $\tau$  can be represented in a  $w$ -dimensional space by a vector  $\overline{H^\delta} = [\langle \overline{b_1}, \overline{h_1} \rangle, \dots, \langle \overline{b_w}, \overline{h_w} \rangle]$ . The  $i^{\text{th}}$  element of  $\overline{H^\delta}$  is calculated by the following equation:

$$\overline{h_i} = \frac{w}{\tau} \sum_{j=\frac{\tau}{w}(i-1)+1}^{\frac{\tau}{w}i} h_j \quad (3)$$

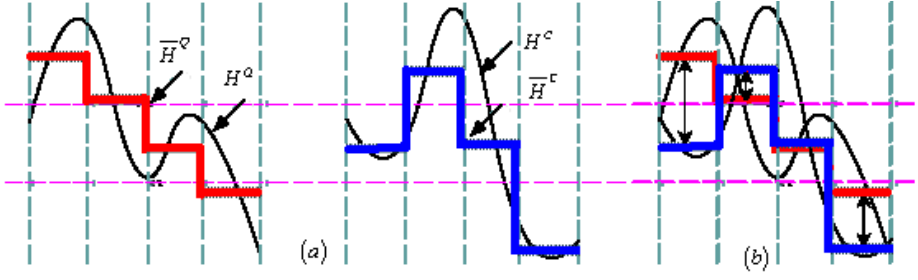
Multi-scale histograms of different time series can be dimension-reduced using different number of line segments, so the approximation can be optimized to describe each multi-scale histogram. Thus, these approximations are defined in a metric domain - this domain does not have a number of dimensions defined.

We developed a new metric distance function in order to quantify the dissimilarity between dimension-reduced multi-scale histograms.

**Definition 1.** The distance function MD ( ) between two dimension-reduced multi-scale time series histograms  $\overline{H^Q} = [\overline{h_1^Q}, \dots, \overline{h_w^Q}]$ ,  $\overline{H^C} = [\overline{h_1^C}, \dots, \overline{h_w^C}]$  is defined as below:

$$d_{reduce}^2 = \frac{\tau}{w} \left( \sum_{j=1}^w (\overline{h_j^Q} - \overline{h_j^C})^2 \right) \quad (4)$$

We note that MD ( ) is a metric, and in particular it obeys the triangle inequality.



**Fig. 1.** Distance between two dimension-reduced multi-scale time series histograms. (a) Two dimension-reduced multi-scale time series histograms  $H^Q$  and  $H^C$ . (b) Steps of the algorithm, exemplifying the distance function calculation.

Figure 1 gives an example of how to calculate the distance between two dimension-reduced multi-scale time series histograms.

Next, we will demonstrate lower bounding characteristic of the distance function  $MD(\cdot)$ . Given two time series  $Q$  and  $C$  of the same length  $n$ , their transformed representations of multi-scale time series histograms  $H^Q = [h_1^Q, \dots, h_\tau^Q]$ ,  $H^C = [h_1^C, \dots, h_\tau^C]$  and further transformed representations of dimension-reduced multi-scale time series histograms  $\bar{H}^Q = [\bar{h}_1^Q, \dots, \bar{h}_w^Q]$ ,  $\bar{H}^C = [\bar{h}_1^C, \dots, \bar{h}_w^C]$ . If  $d_{hist}^2 = (H^Q - H^C)^T$  denotes the distance between two multi-scale time series histograms and  $d_{reduce}^2 = \frac{\tau}{w} \left( \sum_{j=1}^w (\bar{h}_j^Q - \bar{h}_j^C)^2 \right)$  denotes the distance between two dimension-reduced multi-scale time series histograms, then, we have:

**Theorem 1.** With  $d_{hist}$  and  $d_{reduce}$  defined as above, we have that

$$d_{hist}^2 \geq \lambda_\tau d_{reduce}^2 \quad (5)$$

Where  $\lambda_\tau$  is the minimum eigenvalue of  $\Lambda$ ,  $\Lambda$  is determined by  $A = Q^T \Lambda Q$ .

**Proof:** From proof in [7], we know that  $d_{reduce}$  lower bounds the true Euclidean distance, so we have,

$$d_{reduce} \leq \sqrt{\sum_{i=1}^{\tau} (h_i^Q - h_i^C)^2} \quad (6)$$

Since  $A$  is a  $\tau \times \tau$  symmetric matrix and positive semi-definite (get by its definition). Consider the quadratic form  $d_{hist}^2 = (H^Q - H^C)^T A (H^Q - H^C)$ , then it is a theorem of linear algebra that there is a matrix

$$\Lambda = \begin{pmatrix} \lambda_1 & & 0 \\ & \ddots & \\ 0 & & \lambda_\tau \end{pmatrix}$$

(The  $\lambda_i$  in  $\Lambda$  are just the eigenvalues of  $A$ , We may take  $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_\tau$ ) so that

$A = Q\Lambda Q^T$ . Use matrix inequalities, we have

$$\begin{aligned} d_{hist}^2 &= (H^Q - H^C)^T A (H^Q - H^C) = (H^Q - H^C)^T Q\Lambda Q^T (H^Q - H^C) \\ &= \sum_{i=1}^{\tau} \lambda_i \left( q_i^T (H^Q - H^C) \right)^2 \geq \lambda_\tau \sum_{i=1}^{\tau} \left( q_i^T (H^Q - H^C) \right)^2 = \lambda_\tau (H^Q - H^C)^2 \end{aligned}$$

For  $(H^Q - H^C)^2 = \sum_{i=1}^{\tau} (h_i^Q - h_i^C)^2$ , so we have conclusion,

$$d_{hist}^2 \geq \lambda_\tau (H^Q - H^C)^2 = \lambda_\tau \sum_{i=1}^{\tau} (h_i^Q - h_i^C)^2 \geq \lambda_\tau d_{reduce}^2.$$

Based on theorem 1, we can use the distance to prune irrelevant sequences from a database without false dismissals.

Finally, we implemented our proposed dimension-reduced multi-scale time series histograms and the distance MD ( ) using the Slim-tree [8] as the metric access method to answer similarity queries. The Slim-tree has been used because it allows the minimization of the overlap between nodes in the structure as well to measure this overlap.

To speed up search under the metric distance function we will rely on the classic idea of lower bounding. We use Minimum Bounding Omni Region (mbOr) conception and HF-algorithm [9] to elect a set of objects as representative (foci or pivot) of slim-tree, the representative set cardinality of the slim-tree and choose appropriate representative of the slim-tree. The selected representative increases the pruning of distance calculations during the query processing.

## 2.2 Multi-resolution Subsequence Slim-Tree Index Structure

The final part of the paper concentrates on subsequence matching. In contrast to the approaches that they only address the whole sequence matching problem in [5], we propose another dynamic index structure called multi resolution subsequence slim-tree-Index (MSST) that use ideas of the slim-tree index structure and MR-index structure [10]. This index structure stores a grid of subsequence. Each row of this grid corresponds to a different resolution, and is constructed by sliding a window of a pre-specified length on the database sequences. A given query sequence is performed queries corresponding to the resolutions available in the index structure. For a range query, the technique successively reduces the query range by performing an in-

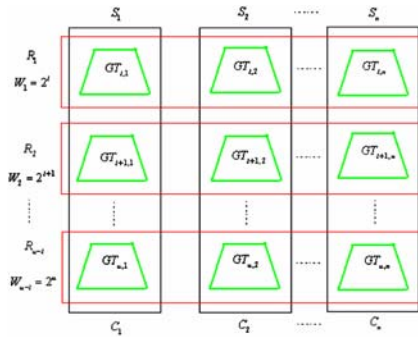
memory search of the queries. This index structure is dynamic and allows queries of arbitrary lengths.

Let  $S$  be the longest time sequence in the databases, where  $2^u \leq |S| \leq 2^{u+1}$  for some integer  $u$ . Similarly, let the minimum possible length for a query be  $2^l$ , for some integer  $u$ , where  $l \leq u$ . Let  $S_1, \dots, S_n$  be the time sequences in the database as shown in Figure 3. Our index structure stores a grid of trees  $GT_{i,j}$ , where  $i$  ranges from 1 to  $u$ , and  $j$  from 1 to  $n$ . Tree  $GT_{i,j}$  is the set of slices for the  $j^{\text{th}}$  subsequence corresponding to window size  $2^i$ .

In order to obtain  $GT_{i,j}$ , we slide a window of length  $2^i$  on sequence  $S_j$ , starting from the leftmost point of  $S_j$ . We transform each sequence of length  $2^i$  into dimension-reduced multi-scale time series histograms, and construct slim-tree index structure from the transformation. This process continues until all subsequences of length  $2^i$  are transformed and constructed.

The  $i^{\text{th}}$  row of the MSST-Index Structure is represented by  $R_i$ , where  $R_i = \{GT_{i,1}, \dots, GT_{i,n}\}$  corresponds to the set of all trees at resolution  $2^i$ . Similarly, the column  $j^{\text{th}}$  of our index structure is represented by  $C_j$ , where  $C_j = \{GT_{1,j}, \dots, GT_{u,j}\}$  corresponds to the set of all trees for the time sequence in the database.

The MSST-Index Structure is dynamic because a row (a data sequence) can be inserted independently. Columns can also be inserted independently; this corresponds to an increase in the size of sequence. Moreover, different rows of the index structure can have different lengths.



**Fig. 2.** Layout of the MSST-index structure.

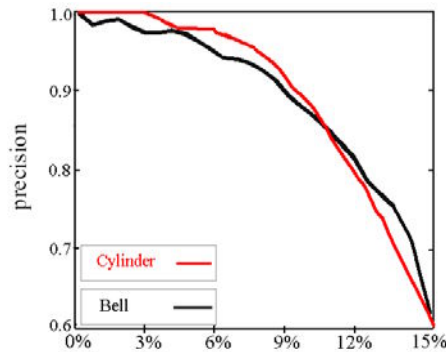
Figure 2 presents layout of the MSST-index structure. The search technique for the MSST-index structure performs a range query for query  $Q$  on the corresponding row

of the index structure at various resolutions available in our index structure. If the length of the query sequence is not a multiple of the minimum window size, then the longest prefix of the query whose length is a multiple of the minimum window size can be used. When all rows at various resolutions available in our index structure have been searched, the disk pages corresponding to the last result set are read. Finally post-processing is carried out to eliminate false retrievals.

### 3 Experimental Results

In this section, we present the results of some experiments that we conducted to evaluate the efficacy of our histogram-based similarity measure and the matching algorithm. All measurements were taken using an Intel Pentium IV 1.5GHz computer running Windows XP. The software was implemented in Borland C++ language.

We first implemented our proposed dimension-reduced multi-scale time series histograms and the distance MD( ) using the Slim-tree as the metric access method to answer similarity queries. Our experimental method is similar to one used in [5]. We first generate 10000 Cylinder-Bell-Funnel (CBF) data sets with 100 examples for each class. Then, we added random Gaussian noise and local time warping to both data sets using a modified program of [5]. The task for our experiment is as follows. Using the shape before the noise is added of a randomly chosen class as a query, retrieve the K most similar members of the same class from the database. We only used cylinder and bell data sets to test for existence queries because of the general shape of bell and funnel are treated as similar. We run the query for 100 times and averaged the results. To evaluate our proposed method, we measure the average precision at the 1% to 15% percent recall points. Figure 3 illustrates the results of dimension-reduced 4-scale time series histograms with 16 bins (based on the previous experimental results of [5], its can achieve reasonable good results). As it can be seen, the retrieval precision is high, the results are always better than 60% of precision, even for queries retrieving 0.5% of the database with 100% of recall.



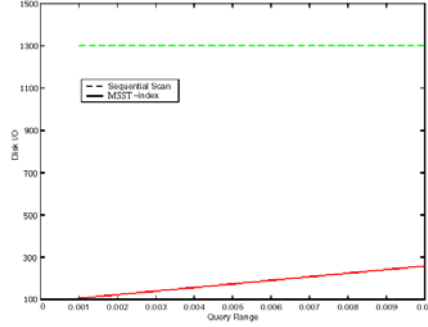
**Fig. 3.** The mean precision of the retrieval method of dimension-reduced multi-scale time series histograms, measured at recall points from 0.5% to 15%.



In our second experiment, we ran an experiment to evaluate its performance of the MSST-index structure to sequential scan concerning space utilization and computational cost. We used a Random Walk dataset generated as follows:

$$x_t = x_{t-1} + z_t \quad (7)$$

Where  $x_0 = 0$ ,  $z_t$  ( $t = 1, 2, \dots$ ) are independent identically distributed (uniformly) random variables in the range  $(-500, 500)$ . This data set contains 1300 sequences of length 1000. We constructed MSST-index structure at resolutions 16, 32, 64, 128, 256, and 512. We performed 1000 arbitrary length random range queries for 10 different values of in  $(0.001, 0.01)$ . Figure 4 plots the average number of disk reads for this experiment. The number of disk reads for the MSST-index structure is about one fifth of that for sequential scan. The MSST-index structure is at least five times faster than seq-scan.



**Fig. 4.** Number of disk reads of the MSST-index structure versus sequential scan for subsequence matching problem.

**Table 1.** Wall-clock time to build the slim-tree using weighted average of time series histogram and dimension-reduced multi-scale time series histograms from the 10000 CBF data sets and the random walk datasets.

Data sets	Histogram	Time(in seconds)
10000 CBF data sets	weighted average of histogram	117.57
	dimension-reduced multi-scale histograms	14.85
random walk dataset (1300 sequences of length 210)	weighted average of histogram	5.05
	dimension-reduced multi-scale histograms	3.08

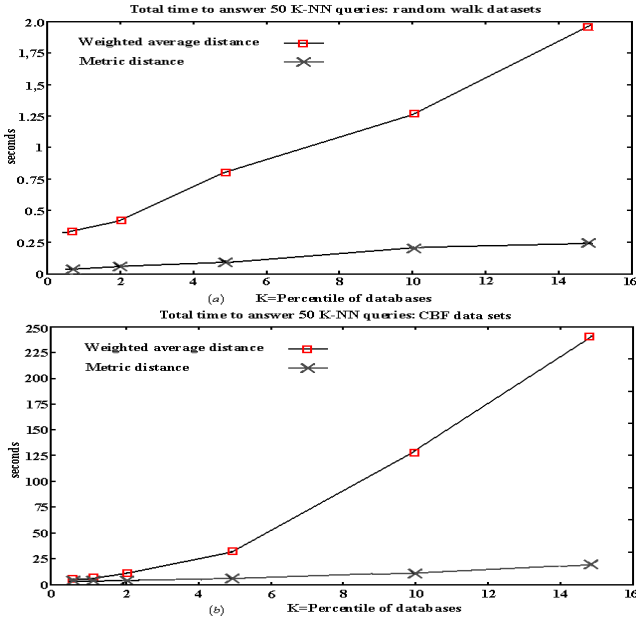
The third experiment is designed to check the time difference for indexing the time series databases through weighted average distance (WAD) used in the weighted average of time series histogram and metric distance (MD) used in dimension-reduced multi-scale histograms and time relevant when answering queries over the weighted average of time series histograms and the dimension-reduced multi-scale histograms.

Table 1 reports the wall-clock times for building the Slim-tree in order to index the weighted average of time series histogram and the dimension-reduced multi-scale histograms.

It can be seen that it is much faster to build the index over the weighted average of time series histogram, i.e. 64% faster in the small dataset, and 790% faster in the larger dataset.

To better understand these results, we also investigated the number of distance calculations performed per second, and found 5, 802,400 metric distances per second and 828,900 weighted average distances per second. That is, the calculation of one metric distance is in average 7 times faster than one weighted average distance over a pair of 256-element arrays.

The largest gain in time when using dimension-reduced multi-scale histograms is to answer queries. Figures 5(a) and 5(b) show the total times needed to answer 50 k-NN queries, when the value of k is specified corresponding to percentage of the database. Thus, the numbers are proportional to the database size and the results can be compared to different database sizes. Figure 5(a) shows the times to answer queries on the random walk datasets of 1300 sequences of length 1000, and figure 5(b) shows the times to answer queries on the 10000 CBF data sets. As it can be seen, the gain ranges from 5 times faster for small k (0.5% of the database) to more than 12 times faster for larger portions of the database (15% of the database).



**Fig. 5.** Total time to answer 50 nearest-neighbors queries for each portion of the dataset: 0.5, 2, 4, 6, 8, 10, 12, 14, and 16%, using the weighted average of time series histogram and dimension-reduced multi-scale histograms. (a) 1300 random walk datasets; (b) 10000 CBF data sets.

## 4 Conclusions and Further Work

We have proposed the dimensionality reduction representation of multi-scale time series histograms, which is performed based on the multi-scale histograms. It is a faster and efficient way to pre-select time sequence in a database and lead to reduce the need of time sequence comparisons when answering similarity queries. A new metric distance function MD ( ) that consistently lower-bounds WED and also satisfies the triangular inequality is also devised. Based on it, we construct the Slim-tree index structure as the metric access method to answer similarity queries. We also extend it to subsequence matching and presented a MSST index structure. Our method guarantees no false dismissal and high search performance and lead to faster and more flexible indexing and retrieval processes. Our experimental result verified the efficacy of our method.

In future work we intend to further increase the speed up of our method by exploiting the similarity of adjacent sequences. Additionally, we also want to investigate the possibility of utilizing the three-level filtering structure by analyzing the CPU and I/O costs of indexing structure proposed in [11].

## References

1. Rakesh Agrawal, Christos Faloutsos, Arun N. Swami.: Efficient Similarity Search In Sequence Databases. Proceedings of the 4th International Conference of Foundations of Data Organization and Algorithms (FODO), Springer Verlag, Chicago, Illinois, (1993)69–84
2. Byoung-Kee Yi, Christos Faloutsos.: Fast Time Sequence Indexing for Arbitrary Lp Norms. In The VLDB Journal, (2000)385–394
3. Eamonn J. Keogh, K. Chakrabarti, M. Pazzani.: Dimensionality Reduction for Fast Similarity Search In Large Time Series Databases. In The VLDB Journal, (1995)490–501
4. Chang-Shing Perng, Haixun Wang, Sylvia R. Zhang, D. Stott Parker.: Landmarks: A New Model for Similarity-Based Pattern Querying In Time Series Databases. In ICDE, (2000) 33–42
5. Lei Chen, M. Tamer Ozsu.: Similarity-Based Retrieval of Time-Series Data Using Multi-Scale Histograms. Technical Report CS-2003-31(Sept 2003)
6. Berman, A., Shapiro, L.G.: Selecting Good Keys for Triangle-Inequality-Based Pruning Algorithms. in Intl. Workshop on Content-Based Access of Image and Video Databases (CAIVD '98), Bombay, India(1998)
7. Keogh, E., Chakrabarti, K., Pazzani, M. & Mehrotra.: Dimensionality Reduction for Fast Similarity Search in Large Time Series Databases. Journal of Knowledge and Information Systems, 263-286
8. Traina, C., Jr., Traina, A.J.M., Seeger, B., Faloutsos, C.: Slim-Trees: High Performance Metric Trees Minimizing Overlap Between Nodes. In Intl. Conf. on Extending Database Technology, Konstanz, Germany (2000)
9. Santos, R.F., Filho, Traina, A.J.M., Traina, C., Jr., Faloutsos, C.: Similarity Search without Tears: The OMNI Family of All-purpose Access Methods. In IEEE ICDE, Heidelberg, Germany (2001)
10. T. Kahveci and A. Singh.: Variable length queries for time series data. In ICDE, Heidelberg, Germany (2001)
11. Ng, R., Tam, D.: Multi-level Filtering for High-dimensional Image Data: Why and How. IEEE Trans. Knowledge & Data Engineering, (December 1999)

# Time Series Prediction Based on Gene Expression Programming\*

Jie Zuo<sup>1</sup>, Chang-jie Tang<sup>1,\*\*</sup>, Chuan Li<sup>1</sup>,  
Chang-an Yuan<sup>1,2</sup>, and An-long Chen<sup>1</sup>

<sup>1</sup> Computer Science Department, Sichuan University,  
Chengdu, Sichuan, China, 610065  
{zuojie,tangchangjie}@cs.scu.edu.cn

<sup>2</sup> Information and technology Department, Guangxi Teachers Education University,  
Nanning, Guangxi, China, 530001

**Abstract.** Two novel methods for Time Series Prediction based on GEP (Gene Expression Programming). The main contributions include: (1) GEP-Sliding Window Prediction Method (GEP-SWPM) to mine the relationship between future and historical data directly. (2) GEP-Differential Equation Prediction Method (GEP-DEPM) to mine ordinary differential equations from training data, and predict future trends based on specified initial conditions. (3) A brand new equation mining method, called Differential by Microscope Interpolation (DMI) that boosts the efficiency of our methods. (4) A new, simple and effective GEP-constants generation method called Meta-Constants (MC) is proposed. (5) It is proved that a minimum expression discovered by GEP-MC method with error not exceeding  $\delta/2$  uses at most  $\log_3(2L/\delta)$  operators and the problem to find  $\delta$ -accurate expression with fewer operators is NP-hard. Extensive experiments on real data sets for sun spot prediction show that the performance of the new method is 20-900 times higher than existing algorithms.

**Keywords:** Gene Expression Programming, Data Mining, Time Series Prediction, Sun Spot Prediction, Differential Equation.

## 1 Introduction

Time series prediction is an important and typical task in data mining. A lot of achievements have been done [1–4]. Recently, George G. S. proposed a good model in time series predication by GA (Genetic Algorithm) [7]. Kang Lishan constructed ordinary differential equations by GP (Genetic Programming) to solve time series prediction problem [8, 9]. However, most existing models are

---

\* Supported by the National Science Foundation of China Grant #60073046, Specialized Research Fund for the Doctoral Program of Higher Education SRFDP #20020610007, The National Science Foundation of Guangxi Grant #0339039, and National Basic Research 973 Program of China 2002CB111504.

\*\* TANG Chang-jie is the associate author.

limited in the complexity of problems. Gene Expression Programming is a revolutionary member in the family of genetic computing introduced by Candida in 2001 [6]. This article proposes two novel methods for Time Series Prediction based on GEP (Gene Expression Programming).

**Problem Specification.** Let  $x(t)$  be a time series. At time point  $t_i = t_0 + \Delta t * i$ , ( $0 \leq i \leq n$ ), the time series data with interval  $\Delta t$  is denoted as

$$X = (x_0, x_1, \dots, x_n) = (x(t_0), x(t_1), \dots, x(t_n)) \quad (1)$$

where  $t_0$  is start time,  $\Delta t$  is time interval. The prediction problem for time series is: To build a model  $T$ ,  $T$  predicts the object data values  $\widehat{x}_m$  at  $t_m$  based on the data set  $\{x_i | i < m\}$ , such that  $|\widehat{x}_m - x_m|$  is as small as possible.

## 2 Slide Window Prediction Method

Sliding Window Prediction Method (SWPM) comes from the following intuition: Given the length of history  $h$ , find a formula  $f$ , such that for any  $m$ , ( $n - h + 1 \leq m \leq n$ ) computing predicting value:

$$\widehat{x}_m = f(x_{m-h}, x_{m-h+1}, \dots, x_{m-2}, x_{m-1}), \quad (h < m \leq n) \quad (2)$$

The difference  $\epsilon$  is as small as possible, where  $\epsilon$  may be absolute difference  $|\widehat{x}_m - x_m|$ , or relative ratio  $|\widehat{x}_m - x_m|/x_m$ , or other measures such as correlation coefficient. The fitness is described in Algorithm 1 based on the input clean assumption.

### Algorithm 1 (Fitness Function for GEP-SWPM Prediction).

```

INPUT : time series z without noise, history length h,
        future length L, formula f to be evaluated
OUTPUT: fitness value for f
for sliding window with length h+L, split z into sequences of segments
FOR each segment g BEGIN
  a = g;
  FOR i = h+1 to h+L BEGIN
    a[i] = f(a[i-h], a[i-h+1], ... , a[i-1]);
  END FOR
  calculate the difference epsilon between a and g
END FOR
calculate the fitness value based on epsilon by error measure function

```

In the last step, there may be various methods to measure the difference  $\epsilon$ , such as absolute error, relative error, and correlation coefficient etc.

**Lemma 1.** *Let  $L$  be the future length in the Algorithm 1. The complexity of Algorithm1 is  $O(L)$ . (Proof omitted.)*

### 3 Mining ODE from Time Series by GEP

The Sliding Window Prediction Method (SWPM) described in Section 2 lacks semantic power. It can not easily uncover the essential properties hidden in the time series. To remedy this shortage, the Ordinary Differential Equation Method is proposed here. The key steps are: (1) Mining high order ordinary differential equations from training data. (2) Predicting by differential equation with respect to initial conditions.

With the transformation  $x_j(t) = x^{(j-1)}(t)$ ,  $j = 1, 2, \dots, m$  we can convert the ordinary differential equations with high rank to first rank

$$x'_m = f_m(t, x_1, x_2, \dots, x_m), \quad x_m(t_0) = x_0^{(m-1)} \quad (3)$$

It is enough to find one formula indicating the essential equation to discover ordinary differential equations with high rank.

By formula 3, the derivative value at  $t_0$  is required. We found that when data contains noise, the small changes in time series data may lead to big changes in differential values. Hence it is not reliable to mine differential equations with very high rank from data with noise. In this paper only differential equations with order less than 3 are considered. To solve the ordinary differential equation, we use the Runge-Kutta algorithm with rank 4. It is accurate enough even with long span. And because the speed is more important, we use the fixed length span in Runge-Kutta algorithm. The complexity analysis of Algorithm 2 is similar to Lemma 1

#### Algorithm 2 (Fitness Function for GEP-DEPM Prediction).

```

INPUT : time series z without noise, derivation value z' of z,
        history length h, future length L, formula f to be evaluated
OUTPUT: fitness value for f.
FOR each sample point t BEGIN
  a[0] = {t, z(t), z'(t)};
  FOR i = 1 to L BEGIN
    solve z''=f(t, z, z') with initial condition in a[i-1] one step,
    get a[i];
  END FOR
  calculate the different epsilon between a and z;
END FOR
calculate the fitness value based on epsilon by error measure function

```

### 4 Differential by Microscope Interpolation (DMI)

To make our model powerful, flexible and tolerant of noise, we propose a brand new method to discover the differential properties from data with noise, called Differential by Microscope Interpolation (DMI). The three-phase actions are “Transformation → Filtering and interpolation under microscope → Inverse transformation”. As is featured with “Zoom in → Differential → Zoom out”, this method is called Differential by Microscope Interpolation (DMI).

### Algorithm 3 (Differential by Microscope Interpolation (DMI)).

INPUT : time series data  $x$  with noise (length  $l$ ), zoom scale  $r$

OUTPUT: data set  $z$  without noise, derivative values  $z'$

- 1) Apply the Fourier transformation to  $x$  and denote the result as  $X$
- 2) Eliminate the high frequency part of  $X$
- 3) Add zeros (the number of zeros is  $r*(l-1)$ ) at high frequency end of  $X$
- 4) Apply Inverse Fourier transformation to  $X$  and denote the result as  $z$
- 5) Magnify each component of  $z$  by  $r$  times
- 6) Computing derivatives values  $z'$  for series  $z$
- 7) Re-sampling on  $z$  and  $z'$  with interval  $r$ , and still denote result as  $z$  and  $z'$
- 8) Return the result  $z$  and  $z'$

**Lemma 2.** *The complexity of Algorithm 3 is  $O((rn)\log(rn))$ . (Proof omitted.)*

## 5 Numeric Constants in GEP

Numeric constants operation is an important problem in both GP and GEP. Some researchers suggest methods with random reproduction. Candida proposed a method in random style [5]. All existing methods in random style are not very efficient in practice.

Constant technique is the right technique bringing simplicity to GEP. With the powerful structural optimization of GEP, we propose a new approach called Meta-Constants Method to improving numeric, and apply it to our GEP-SWPM and GEP-DEPM methods. The key points in GEP constant technique for time series prediction are the following: (1) Giving a candidate set  $C$  of constants according to expert knowledge of the specific domain. (2) GEP tries to mine a valid expression  $E = E(c_1, c_2, \dots, c_n)$ , where constants  $c_i$  in  $C$ , are operators in a function set such as  $\{+, -, *, /, \sin, \cos, \dots\}$  and the subscript of operators in  $E$  does not exceed a fixed number  $h$ , i.e. the head length of GEP. One interesting question arises that how powerful the mined result  $E$  would be. Does it cover a wide enough range? Is it dense enough (accurate enough)? Theorem 1 shows that with only two operators in the function set, our Meta Constant Method is powerful enough in the sense that the generated set can cover the interval  $[\delta, \delta(1/2)(3^{n+1} - 1)]$  with interval  $\delta$ . In practice, more operators can be added to the function set, and therefore, we will get wider range and higher accuracy.

**Theorem 1.** *Let the operators set be  $Op = \{+, -\}$   $E$  be a valid expression with at most  $n$  operations from  $Op$  (operators may be repeated).  $C$  be the candidate constant set defined as*

$$C = \{c_i = \delta 3^i \mid 1 \leq i \leq n \wedge i \in \mathbb{Z}\} \quad (4)$$

where  $\delta$  is a constant.  $\mathbb{Z}$  denotes the integer set. and  $MC(c_1, c_2, \dots, c_n)$  be a constant generated by our meta-constant model. Then  $E = \{MC(c_1, c_2, \dots, c_n) \mid c_i \in C\}$  covers the set  $\{\delta + k * \delta \mid k \in \mathbb{Z}\} \cap [\delta, \delta(1/2)(3^{n+1} - 1)]$

*Proof.* We prove it by induction on  $n$ . When  $n = 1$ ,  $\delta(1/2)(3^{n+1} - 1) = 4\delta$ , clearly,  $d_1 = c_1 = 1\delta$ ,  $d_2 = c_2 - c_1 = 3\delta - 1\delta = 2\delta$ ,  $d_3 = c_3 = 3\delta$ ,  $d_4 = c_1 + c_2 = 3\delta + 1\delta = 4\delta$ , so, all the four values can be expressed by valid expressions with  $n = 1$  operations, as desired. Suppose  $n = k$  and all target values can be expressed by valid expression with at most  $n$  operations. Consider the case  $n = k + 1$ . Four cases arise:

1.  $1 \leq i \leq (1/2)(3^{k+1} - 1)$ : By the assumption of induction,  $d_i = i * \delta$ ,  $d_i$  can be expressed by a valid expression with operations not exceeding  $k + 1$ .
2.  $i = 3k + 1$ :  $d_i$  is in  $C$  and can be expressed without operators.
3.  $(1/2)(3^{k+1} - 1) + 1 \leq i \leq 3^{k+1} - 1$ : Let  $x = (1/2)(-1 + 3k + 1)$ , that is, we express the range as  $x + 1 \leq i \leq 3^{k+1} - 1 = 2x - 1$ . Let  $j = x - (i - x) + 1$ , since  $x + 1 \leq i$ , hence  $1 \leq i - x$ , therefore,  $j \leq x$ . Note that  $i \leq 2x - 1$ , hence  $1 \leq j$ , that is  $1 \leq j \leq x = (1/2)(3^{k+1} - 1)$ . By induction assumption,  $d_j = j * \delta$ . Let  $d_i = \delta 3^{k+1} - d_j = \delta(3^{k+1} - j) = \delta(3^{k+1} - ((3^{k+1} - 1) + 1 - i)) = i * \delta$ . Note that,  $\delta * 3^{k+1}$  is in candidate set. By the construction, it is enough to add only one operation. It can be expressed with operations not more than  $k + 1$ .
4.  $3k + 1 + 1 \leq i \leq 3k + 1 + (1/2)(-1 + 3k + 1) = (1/2)(-1 + 3k + 2) = (1/2)(-1 + 3n + 1)$ : The proof is similar and can be finished by replacing “+” with “-”.

To introduce the significant Theorem 2, we need the following concepts and notations.

**Definition 1 (Minimum Accurate Expression).**

1. Let  $I$  be an interval of real numbers,  $\delta$  be a positive number (to measure accuracy).  $C$  be the Meta constant set,  $F$  be the set of valid operators.  $E = E(c_1, c_2, \dots, c_k)$  be a valid expression with operators selected from  $F$  and constants from  $C$ .  $E$  is said to be the  $\delta$ -accurate expression for  $I$ , if for each  $v$  in  $I$ , there exists  $c_1, c_2, \dots, c_k$  in  $C$  such that  $|v - E(c_1, c_2, \dots, c_k)| \leq \delta/2$ .
2. Let  $E_1$  and  $E_2$  be  $\delta$ -accurate expressions for  $I$ .  $E_1$  and  $E_2$  are said to be equivalent, and denoted as  $E_1 \sim E_2$ , if for any parameters  $c_1, c_2, \dots, c_k$ ,  $E_1(c_1, c_2, \dots, c_k) = E_2(c_1, c_2, \dots, c_k)$ .
3. The operator numbers of  $E_1$  is denoted as  $\#(E_1)$ .
4. Expression  $E$  for  $I$  is said to be minimum if  $\#(E) = \min\{\#(E) \mid E \sim E_1\}$ .

Clearly, the minimum expression is with simplest structure and hence with low cost of computation. The minimum expression may not be unique.

**Theorem 2.** Let  $I$  be an interval  $[s, t]$  of real numbers with length  $L$ .  $E$  be a minimum  $\delta$ -accurate expression for  $I$  discovered by GEP-MC method. Then,  $\#(E) \leq \log_3(2L/\delta)$ .

*Proof.* It is enough to show that we can construct an expression  $E$  with  $s$  operators,  $s \leq \log_3(2L/\delta)$ , and  $E$  can express each  $v$  in the set  $\{s + i * \delta \mid s + i * \delta \leq t\}$  with error less than  $\delta/2$ .



Let  $N = \lceil L/\delta \rceil$ ,  $M = \min\{K \mid K \in \mathbb{Z} \wedge K \geq \log_3(2N + 1) - 1\}$ . Hence  $M \geq \log_3(2N + 1) - 1$ , that is,  $3M + 1 \geq 2N + 1$ , thus  $(1/2)(3M + 1 - 1) \leq N = L/\delta$ . According to Theorem 1, in our MC-method, we can use the operator set  $\{+, -\}$  as function set, and use  $\{\delta * 3^i \mid 0 \leq i \leq M\}$  as candidate constants set. By the conclusion of Theorem 1, we can construct an expression satisfying following conditions:

1. for each value  $v$  in  $[s, t]$ ,  $|v - E(c_1, c_2, \dots, c_n)| \leq \delta/2$  for some constants  $c_1, c_2, \dots, c_n$ .
2.  $\#(E) \leq M$ .

In summary, to express each  $v$  in the set  $\{s + i * \delta \mid s + i * \delta \leq t\}$  with error less than  $\delta/2$ . we need at most  $K$  operators,  $K = \lceil \log_3(2(L/\delta) + 1) - 1 \rceil$ . This completes the proof.

The Theorem 2 signifies that it theoretically proves following conclusion: Our Meta-Constants Method is simple enough, powerful enough and accurate enough. We have applied these techniques to our GEP-SWPM and GEP-DEPM methods. The experimental results on real data sets of Sun Spot prediction will verify this conclusion.

**Theorem 3.** *Let  $I = [s, t]$  be an interval of real numbers,  $\delta$  be a positive number. The candidate constants set  $C$  is other than formula 4. The problem to find a  $\delta$ -accurate expression for  $I$  with operator less than  $\log_3(2L/\delta)$  base on  $C$  is NP-hard.*

*Proof.* Let the length of  $I$  be  $L$ . Let  $E$  be the minimum  $\delta$ -accurate expression for  $I$ . The Theorem 2 asserts that the number of operators in  $E$ , i.e.  $\#(E)$  not exceeds  $K = \log_3(2L/\delta)$ . Since here the number of operators is less than  $K$ , and the candidate constants set  $C$  is other than formula 4, the construction in Theorem 2 can not be applied. As the technique described in [11], then it can be solved do the following algorithm:

1. Let  $R$  be the set of valid expression of at most  $K - 1$  operators.
2. Clearly the size of  $R$  is greater than in there order of  $O(2^{K-1})$ .
3. Make lucky guess, that is, enumerating a expression  $E$  from  $R$  non-deterministically.
4. Calculate  $\epsilon = |E - d|$ , where  $d \in I$  is the target value.
5. Return true if  $\epsilon \leq \delta$ .
6. Repeat the procedure, and return false when all  $E \in R$  has being checked.

Note that the step (4) needs  $K - 1$  operations, where  $K = \log_3(2L/\delta)$ . Hence step (4) to (6) can be finished in polynomial time. Note that step (1) is non-deterministic. Hence the whole computing can be complete in polynomial time by non-deterministic Turing machine. This completes the proof.

Note that, non-deterministic mechanisms are hidden in evolution computing and GEP has high performance, hence GEP can solve NP-hard problem with high successful rate in acceptable speed.

## 6 Test and Evaluation

As a part of an integrated research work, the experiments for this article are completed on our platform called zGEP. We have spent about 8 months to implement zGEP by Java. It is flexible, extendable and transplantable. The GEP terminologies and notations in the experiment report are as that in [5]. By the limitation of pages, only four main experiments are presented here.

The test environment is as following. CPU: AMD Athlon XP 1800+, Memory: 512MB, OS: Windows 2000 Professional, Platform: zGEP, Mathematica, Test data: Sun spot data [7] and synthesized data.

### Experiment 1 on Numeric Constants

This experiment is designed to test the performance of our numeric constants technique MC (meta-constant) in time series prediction by GEP. The Experiment data is synthesized by formula of circle  $s = \pi r^2$ . The experiment shows, the MC method is powerful. It can be finished in a few seconds with high accuracy. The parameters of GEP are as Table 1, where “!” in function set denote negative operator, and “0, 1,  $\dots$ , 9” in terminal set denote the indexes in constant set. The rate of any two nearest constants in constant set is golden rate.

**Table 1.** Experiment parameters.

Time limit	10s	Selection operator	tournament
Function set	+ - * / !		with size 4
Terminal set	$r, 0, 1, \dots, 9$	Mutation rate	0.044
Constant set	0.1315, 0.2128,	One-point recomb rate	0.3
	0.3443, 0.5571,	Two-point recomb rate	0.3
	0.9015, 1.4588,	Gene recomb rate	0.1
	2.3605, 3.8195,	IS transposition rate	0.1
	6.1804, 10.0007	IS elements length	1,2,3
		RIS transposition rate	0.1
Population size	50	RIS elements length	1,2,3
Number of genes	3	Gene transposition rate	0.1
Head length	6		

Run 100 times, it gives the value of  $\pi$  with accuracy higher than 0.995.

$$s = (((r) * (r)) + ((r) * (((2.3605) - (0.9015)) * ((1.4588) * (r)))))) + (((0.1315) * ((r)/(0.5571))) * (((r)/(3.8195)) * (0.2128))) \quad (5)$$

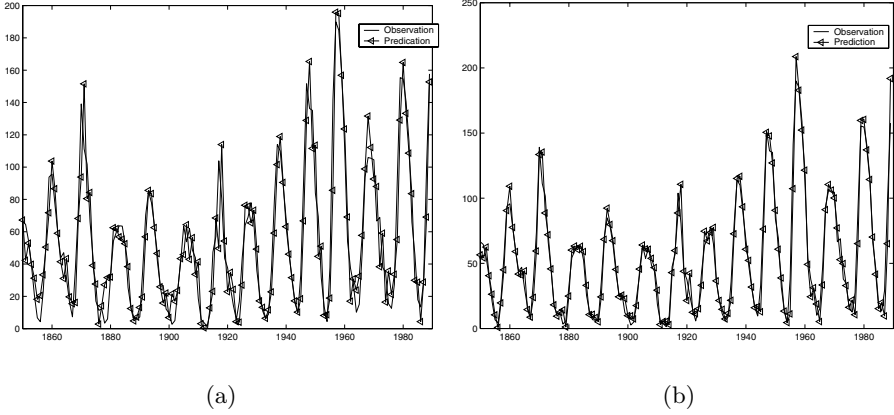
By Mathematica, it is transformed into an accurate enough result:  $s = 3.14154r^2$ .

### Experiment 2 Predict Sun Spot by GEP-SWPM

The Sun Spot Time Series is a benchmark to test the algorithm for time series prediction [7]. We use the real data of sun spot series. The parameters are as Table 2. Ran experiment 100 times, and get average fitness 0.9023. Figure 1 (a)

**Table 2.** Parameters for GEP-SWPM.

Training data	from 1850 to 1979	History length	12
Test data	from 1980 to 1989	Fitness function	Correlation coefficient
Time limit	20s	Constants process	By our MC method
Number of genes	5	Differential Process	By our DMI method
Gene head length	10	The other parameters are same as that in Table 1	

**Fig. 1.** (a) The prediction result of GEP-SWPM method. (b) The prediction result of GEP-DEPM method.

Shows that the method is rather accurate in sun spot prediction. One of mined formula is

$$\begin{aligned}
 x_m = & (((((-(x_{m-2}) - ((x_{m-1}) + (-(x_{m-10})/(x_{m-4})))))) \\
 & / (3.8195)) + ((x_{m-1}) + (0.3443))) + (((-(x_{m-4}) - (x_{m-1}))) \\
 & / (x_{m-12}))) + (((-(x_{m-3}))/ (3.8195)) / (-(x_{m-9}) \\
 & * (0.5571)))) + (((-(x_{m-2}) - (x_{m-9}))) / (3.8195))
 \end{aligned} \quad (6)$$

After equivalent transformation by Mathematica, it is transformed into:

$$\begin{aligned}
 x_m = & 0.3443 + 0.2618 * x_{m-9} - (0.2618 * x_{m-10}) / x_{m-4} \\
 & - x_{m-4} / x_{m-12} + (0.4700 * x_{m-3}) / x_{m-9} \\
 & - 0.5236 * x_{m-2} + 1.2618 * x_{m-1} + x_{m-1} / x_{m-12}
 \end{aligned} \quad (7)$$

Compared with the research result reported in [10], our method is 900 times faster than that in [10].

### Experiment 3 Predict Sun Spot by DEPM

This experiment tests the performance of our DEPM method on sun spot data set. The GEP parameters are as following: Time limit: 30s, Number of genes: 3, Gene head length: 6, Function set: + - \* / ! log sin cos, The other parameters are same as that in Table 2. We hope mine an Ordinary Differential

Equation between the amount of sun spot and the number of year in A.D. (such as 1998 A.D.) The experiments are repeated 100 times. The average correlation coefficient is as high as 0.9214. Figure 1 (b) shows the prediction result of one running. The Differential Equation mined from data is

$$x''(t) = (((((10.0007) - (x(t))) - ((x'(t)) - (10.0007)))/(3.8195)) + ((\cos((x(t)) + (x(t))))/((\sin(0.5571))/(3.8195)))) + (\log(((t) + (0.9015)) + (t))) \quad (8)$$

After equivalent transformation by Mathematica, it is transformed into

$$x''(t) = 5.2367 + 7.2240 * \cos(2 * x(t)) + \log(0.9015 + 2 * t) - 0.2618 * x(t) - 0.2618 * x'(t) \quad (9)$$

The experiment shows that the prediction results are very close to the observation data. The research work reported in [9] uses Differential Equation mined by GP takes one hour to complete the computation while our method takes 30 seconds. Hence, our method is 50 times faster than the method using GP.

## 7 Conclusion

Time series prediction is an important and typical task in data mining. Based on GEP (Gene Expression Programming), two methods for prediction of time series are proposed, namely, SWPM and DEPM.

The time series prediction by GEP is just a beginning; a lot of work remain to be done, such as multi-gene, the application of evolutionary dynamic, etc.

## References

1. Jiawei Han, WanGong and Yiwen Yin: Mining Segment-wise Periodic Pattern in Time Related Databases, Proc. Of 1998 of International Conf. On Knowledge Discovery and Data Mining(KDD'98) New York City, NY. Aug 1998.
2. Ozden,B., Ramaswamy, S., and Silberschatz, A.: Cyclic association rules, In Proc. Of 1998 international Conference Data Engineering ICDE'98, p412-421. 1998.
3. Xiang JT, Du JQ, Shi J: Dynamic Data Processing: Time Series Analysis. Meteorology Press, 1988.
4. Gan Renchu: The Statistical Analysis of Dynamic Data. Beijing University of Science and Technology Press, 1991.
5. Candida Ferreira: Gene Expression Programming: A new adaptive Algorithm for solving Problems. Complex Systems Vol. 13, Nu. 2, pp87-129, 2001.
6. Zuo Jie, Tang Changjie, Zhang Tianqing: Mining Predicate Association Rule by Gene Expression Programming. WAIM02 (International Conference for Web Information Age 2002), Springer Verlag (LNCS) Vol.2419, pp.92-103.
7. George, G. S.: Forecasting Chaotic Time Series with Genetic Algorithms. Physical Review E, 1997, 55(3): 2557-2567.
8. Kang LS, Cao HQ, Chen Y: The Evolutionary Modeling Algorithm for System of Ordinary Differential Equations. *Chinese Journal of Computer*, 1999, 22(8): 871-876.

9. Cao Hongqing, Kang Lishan, Chen Yuping: The Evolutionary Modeling of Higher-Order Ordinary Differential Equations for Time Series Analysis. *Mini-Micro System*, Vol.21, 2000.4
10. Li Minqiang, Kou Jisong: The Theory and Application of Genetic Algorithm. ISBN 7-03-009960-5/C.67. Science Press.
11. Michel Sipser: Introduction to the theory of computing. PWS publishing company, a division of Leaning, 1997.

# A Grid-Based Index Method for Time Warping Distance

Jiyuan An<sup>1</sup>, Yi-Ping Phoebe Chen<sup>1,2</sup>, and Eamonn Keogh<sup>3</sup>

<sup>1</sup> School of Information Technology, Faculty of Science and Technology  
Deakin University, Melbourne, VIC 3125, Australia  
{jiyuan, phoebe}@deakin.edu.au

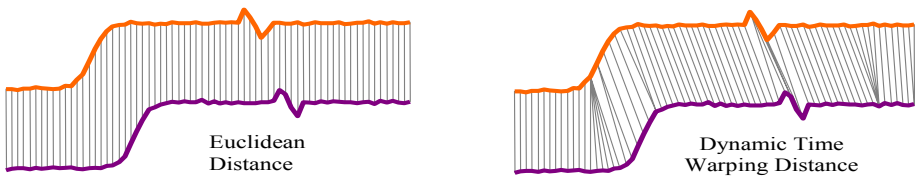
<sup>2</sup> Australian Research Council Centre in Bioinformatics

<sup>3</sup> University of California-Riverside, Riverside, CA 92521, USA  
eamonn@cs.uci.edu

**Abstract.** Recently DTW (dynamic time warping) has been recognized as the most robust distance function to measure the similarity between two time series, and this fact has spawned a flurry of research on this topic. Most indexing methods proposed for DTW are based on the R-tree structure. Because of high dimensionality and loose lower bounds for time warping distance, the pruning power of these tree structures are quite weak, resulting in inefficient search. In this paper, we propose a dimensionality reduction method motivated by observations about the inherent character of each time series. A very compact index file is constructed. By scanning the index file, we can get a very small candidate set, so that the number of page access is dramatically reduced. We demonstrate the effectiveness of our approach on real and synthetic datasets.

## 1 Introduction

Similarity search in time series databases is an important application of Spatial Access Methods (SAMs) [1, 8, 13, 14, 16, 22, 23]. In order to map time series into a metric, indexable space, Euclidean distance is widely used. However, there is an increasing awareness that Dynamic Time Warping (DTW) distance reflects the real similarity of time series better than Euclidean distance [13, 22, 23]. This is because DTW is less sensitive to discrepancies on time axis, allowing similar shapes to match even when out of phase. Figure 1 shows that two subjectively similar subsequences can have a large Euclidean distance, but that DTW, by aligning peak-to-peak, and valley-to-valley, can discover the true underlying similarity.

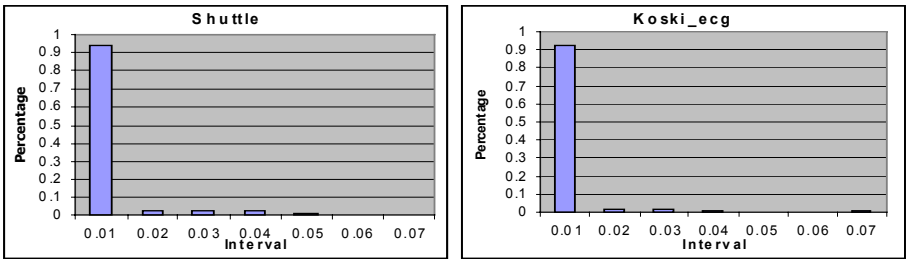


**Fig. 1.** A visual comparison of Euclidean distance and Dynamic Time Warping. By allowing flexibility in the time axis, Dynamic Time Warping achieves a more intuitive alignment between the two time series.

To index time series with a multidimensional index structure, each data point on the time series corresponds to a coordinate of dimensional space. Every time series therefore becomes a single point in high dimensional space and similar time series will have a small distance in this space. However, the space usually has very high dimensionality, in the order of hundreds or thousands of dimensions. With high dimensional data, the R-tree index structure cannot prune many objects, and all the nodes in the tree structure will be checked. This reduces the R-tree’s effectiveness to worst than that of simple sequential scan [4, 6], because the R-tree’s random access is very ineffective comparing with sequential access. To utilize R-tree index structure effectively, dimensionality reduction is necessary. DFT, SVD and DWT are the classic methods [8, 16]. Recently, other techniques have been introduced, including PAA and APCA (Adaptive Piecewise Constant Approximation) [14]. All these are based on Euclidean distance. However, as demonstrated above, Euclidean distance has a fatal defect: it is very brittle when confronted with variability in the time axis.

Unfortunately DTW does not satisfy triangle inequality [22], and virtually all SAM’s assume this property. Nevertheless several techniques for indexing methods of DTW have recently been proposed [13, 23] by introducing the global constraints on time warping. Keogh successfully applied the R-tree index structure to DTW [13] by using global constraints, i.e. the warping path is limited in Sakoe-Chiba Band or the Itakura Parallelogram.

In this paper, we propose a radically different index method, motivated by observations about typical time series. Each time series has its own character; a data point in time series has some relationship with its neighbors. We can visually illustrate the difference of adjacent data points on a time series below. First we normalized the all data points in the time series to  $[0, 1]$ , and then calculated the difference with their proceeding data points. Figure 2 shows the histograms of the differences of two real datasets, *Shuttle* and *Koski\_ecg* [15]. From this figure, we can see that more than 90% of the data have less than 0.01 differences with their previous data points. This motivates us to exploit this fact by proposing a new dimensionality reduction technique, which we introduce in Section 3.



**Fig. 2.** Two real datasets (*Shuttle*, *Koski\_ecg*) histogram. More than 90% data points have less than 0.01 difference with previous data points.

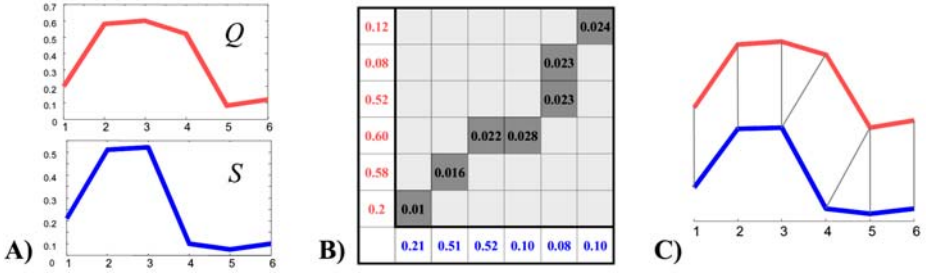
Using our dimensionality reduction method, we construct an index structure for time warping distance. In Section 2, we briefly describe relevant background material. The new dimensionality reduction method and indexing technique are described in the Section 3. In the Section 4, we show experimental results. We offer some conclusions in Section 5.

## 2 Background

Euclidean distance is the simplest method to measure the dissimilarity of time series as shown in Figure 1(left). Equation 1 gives formula, where  $Q$  and  $S$  are two time series.  $Q = (q_1, q_2, \dots, q_n)$ ,  $S = (s_1, s_2, \dots, s_n)$ .

$$D(Q, S) = \sqrt{\sum_{i=1}^n (q_i - s_i)^2} \quad (1)$$

Note that Euclidean distance explicitly maps the  $i^{\text{th}}$  point in one time series to the  $i^{\text{th}}$  point in another. In contrast, DTW can achieve a nonlinear mapping of points, including allowing one-to-many and many-to-one mappings. Dynamic time warping distance discovers the warping path that allows the distance between two time series have smallest value. Figure 3 shows two time series sequences  $Q = (0.2, 0.58, 0.6, 0.52, 0.08, 0.12)$ ,  $S = (0.21, 0.51, 0.52, 0.1, 0.075, 0.1)$ . They are subjectively similar time series. Their dynamic time warping distance is 0.153, but the Euclidean distance is 0.433, suggesting they are very dissimilar.



**Fig. 3.** An example of a DTW calculation. A) Two time series can be used to construct a matrix containing the differences between each point in one time series and every point the other time series. B) A warping path moves from one diagonal corner to another attempting to minimize the cumulative sum of values along the path. C) The resulting alignment.

Like Euclidean distance, Dynamic time warping can also be given with a simple formula. We denote  $Q(q_1, q_2, \dots, q_n)$  and  $S(s_1, s_2, \dots, s_n)$  as two time series. We assume  $\overline{pre}(Q) = \{q_1, q_2, \dots, q_{n-1}\}$ ,  $\overline{last}(Q) = q_n$ ,  $\overline{pre}(S) = \{s_1, s_2, \dots, s_{n-1}\}$ ,  $\overline{last}(S) = s_n$ , the DTW can be obtained by:

$$DTW^2(Q, S) = D^2(\overline{last}(Q), \overline{last}(S)) + \min \begin{cases} DTW^2(Q, \overline{pre}(S)) \\ DTW^2(\overline{pre}(Q), S) \\ DTW^2(\overline{pre}(Q), \overline{pre}(S)) \end{cases} \quad (2)$$

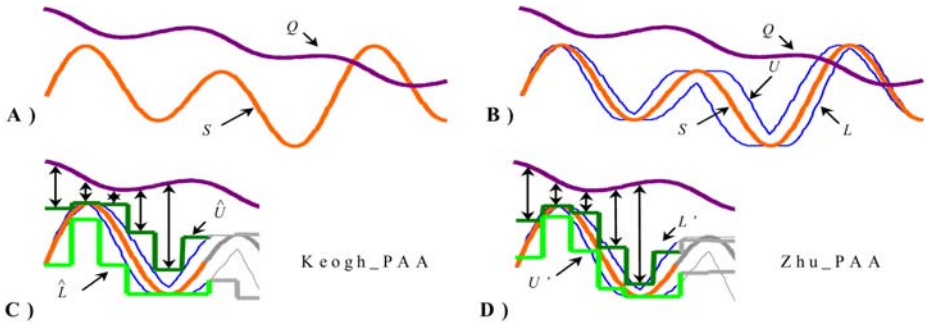
The procedure proceeds recursively on the initialization of  $DTW(\Phi, \Phi) = 0$ . This review is necessarily brief; we refer the interested reader to [13, 23] for a more detailed explanation.



## 2.1 Related Work

We will briefly review the recently introduced indexing methods for dynamic time warping [23,13]. They are based on the observation that if it is possible to tightly lower bound a distance measure, and then it is nearly always possible to index it. In [13] the author introduced the first tight lower bound for DTW. The technique is based on the idea of building a “protective” envelope around a candidate time series and showing that the Euclidean distance between the query and the nearest part of the envelope is a true lower bound for DTW, see Figure 4 for a visual intuition. While this lower bound allows fast sequential search it does not directly solve the high dimensionality problem, however it does transform the problem in such a way that the classic dimensionality reduction techniques can be used. While any orthogonal transformation can be used (i.e DFT, DCT, DWT, SVD), the Piecewise Aggregate Approximation (PAA) is used for simplicity [16].

In the original work, Keogh’s method used PAA to bind the envelope by minimum and maximum values. Zhu’s minor extension is to use average of the upper or lower envelope, which is narrower, and thus a tighter bound than that in Keogh’s method. However, in Section 4, our experiments show Zhu\_PAA becomes as weak a lower bound as Keogh\_PAA when warping width increases.



**Fig. 4.** A) A query  $Q$  and a candidate sequence  $S$ . B) An envelope  $\{L, U\}$  can be built around  $S$ , it has been shown that the Euclidean distance between  $Q$  and the envelope is a lower bound for  $DTW(Q, S)$ . C) To index this envelope in an R-tree, Keogh suggested segmenting it, and Zhu noted that an even tighter form of segmentation is possible as shown in (D).

## 3 Our Approach

Our approach is based on the concept of *Grid*. In our approach, the horizontal axis of a time series is divided into slices. The vertical axis is also divided for representing values of data points by quantization. As noted in Section 2, unlike other data types, data points on the time series typically are highly correlated with its previous/following data points, a phenomenon known as *autocorrelation*. Most data points have very similar values with their adjacent data points. Our dimensionality reduction method, which we call *Grid*, has a very compact representation of time series. It omits the data points that have very similar value with its previous data value. In this respect it is very similar to the relatively obscure *boxcar* method of dimensionality reduction [10],

a technique little known outside the chemical process industry. We will define our representation in more detail in Section 3.1.

The similarity search in our approach is performed in two phases. In phase 1 (*filtering*), the compressed file is scanned. The access method in this phase is sequential access which is much more efficient than random access. The factor is known to be 10 [20], i.e. if a tree index structure has less than 1/10 pruning power, scanning the index file is faster than using the tree index structure. In phase 2 (*refinement*), calculating the exact distances between the query with candidates obtained in phase 1. In this phase the access method utilizes random access. However, because of the small size of candidate set, the time of access to data file is very limited. As we will empirically show, the total number of disk access in our approach is much fewer than that for tree index structures.

### 3.1 Grid Dimensionality Reduction and Data Representation

We approximate each time series by a set of segments of varying lengths. The number of segments is not a parameter, but is adaptively determined by the time series itself. If a time series is a simple shape, only a few segments are used to represent it. In the pathological case of a constant line, we only use a single segment. Many segments are necessary for a complex time series. This adaptiveness ensures that the information lost in target domain is minimal, and a good lower bound is obtained for similarity search. We describe our dimensionality reduction method below.

First, we normalize the data space into the range of  $[0, 1]$ , then the normalized values are quantized. Second, according to the error tolerance  $\mathcal{E}$  (which is a parameter), omittable data points are determined. Since the omitted data point positions are not identical between different time series, we must record their position. A worked example will aid clarity.

Given a time series  $(v_1, v_2, \dots, v_n) = (0.55, 0.61, 0.49, 0.81, 0.74, 0.63, 0.41, 0.48)$ , it is approximated in the following 2 steps.

1. **Quantization:** The time series is quantized to  $2^b$  intervals in the vertical axis, where  $b$  is the number of bits used for approximating  $v_i$ . In the example, if we assumed that  $b = 3$ , then by  $\alpha_i = \lfloor v_i \times 2^b \rfloor$ , the quantized time series of the example becomes

$$(\alpha_1, \alpha_2, \dots, \alpha_8) = ((100)_2, (100)_2, (011)_2, (110)_2, (101)_2, (101)_2, (011)_2, (011)_2)$$

2. **Reduction:** If a value of a data point is close to that of its previous data point, then it is omitted. We use a tolerance parameter  $\mathcal{E}$ . Whether a data point is omitted or not is determined by Equation (3).

$$\alpha_{i+1} = \begin{cases} \text{omitted} & -\mathcal{E} \leq v_{i+1} - \alpha \times h \leq h + \mathcal{E} \\ \lfloor v_{i+1} \times 2^b \rfloor & \text{otherwise} \end{cases} \quad (i=1, 2, \dots, n) \quad (3)$$

In the Equation 3,  $h$  is the height of grid ( $=1/2^b$ ) and  $\alpha$  is the representative value of the segment. If  $\alpha_i$  is the first data point of a segment, then we have  $\alpha = \alpha_i$ .

That is, the representative value of each segment is the quantized value of the first data point in the segment.

To reduce the dimensions, the range of error tolerance is added as shown in Figure 6. The data points in a gray rectangle belong to a single segment, so their values are omitted except the first one. In this example,  $\mathcal{E}$  is set to  $h/2 = 1/16$ , and after the reduction, the time series becomes:

$$(\alpha_1, \alpha_2, \dots, \alpha_8) = ((100)_2, \times, \times, (110)_2, \times, (101)_2, (011)_2, \times)$$

where “ $\times$ ” denotes the omission of the data points. The details are described below.

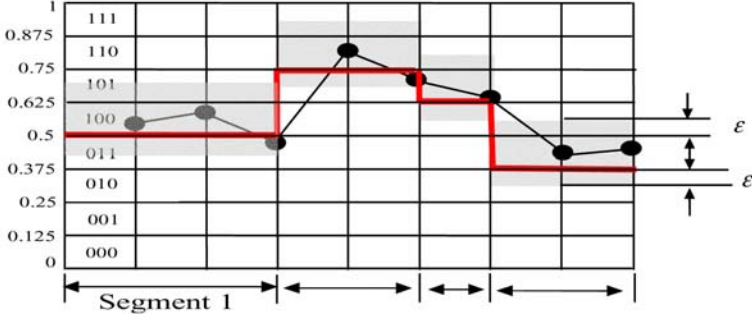
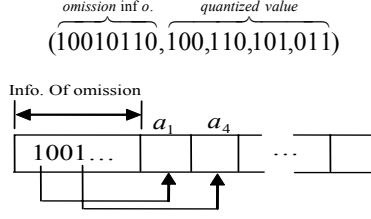


Fig. 5. Grid dimensionality reduction. The thick line approximates the time series.

By Equation 3,  $v_1$  is out of range  $[-\mathcal{E}, \mathcal{E} + h] = [-1/16, 3/16]$  ( $\alpha$  is initialized to 0), so the first data point  $\alpha_1$  must be stored. Now we have  $\alpha = \alpha_1 = (100)_2$ . Using Equation 3, we can easily find that the second point  $\alpha_2$  should be omitted. This is because  $v_2 - \alpha \times h = 0.61 - 4 \times 0.125 = 0.11$  is in the range of  $[-1/16, h + 1/16] = [-1/16, 1/8 + 1/16]$ . Analogously, we can determine whether  $v_3$  can be omitted,  $v_3 - \alpha \times h = 0.49 - 4 \times 0.125 = -0.1$  is in the above range, thus  $\alpha_3$  is also omitted. However,  $v_4 - \alpha \times h = 0.81 - 4 \times 0.125 = 0.31$  exceeds the range, so  $\alpha_4$  must be stored. The rest of the time series is processed in a similar way. Because the segments are variable-length, it is not known in advance how many we will have and which of the data points have been omitted. To record this information, we use an  $n$ -bit length bit pattern  $\beta$  which is associated with the approximation of a time series data. If  $\alpha_i$  is omitted, the corresponding bit  $\beta_i$  is set to 0. Otherwise it is set to 1. The structure of an entry representing one time series data is shown in Figure 6. In the case of the example above, the entry representing the series data becomes:

### 3.2 KNN-Search of DTW

Each time series is represented by a reduced data which consists of index file. In the filtering phase, we calculate the lower bound between query and time series. In our previous work [3], we have shown a lower bound of Euclidean distance in Grid-based dimensionality reduction. We can prove that the lower bound of DTW can be calculated using lower bound of Euclidean distance.



**Fig. 6.** Representation of time series for Grid dimensionality reduction. An  $n$ -bit binary string denotes which data points in  $v$  are omitted. The following boxes indicate quantized value of data points.

**Observation.** By using lower bound of Euclidean distance, we can change Equation 2 as below.

$$DTW_{LB}^2(Q, S) = D_{LB}^2(\overline{last}(Q), \overline{last}(S)) + \min \begin{cases} DTW_{LB}^2(Q, \overline{pre}(S)) \\ DTW_{LB}^2(\overline{pre}(Q), S) \\ DTW_{LB}^2(\overline{pre}(Q), \overline{pre}(S)) \end{cases}$$

Where  $D_{LB}$  is a Euclidean distance to calculate the distance between two data points.

It is not difficult to understand from the Figure 6. Since each lattice lower bounding its DTW, the value of lattice in the top-right lower bounding the DTW as well. In this paper we omit description how to calculate  $D_{LB}$ . Please refer [3] for details.

```

Algorithm KNNSearch(Q, K)
1.   Foreach entry S in index file
2.     compute  $DTW_{LB} < k$  or  $DTW_{LB} < result[k].DTW$ 
3.     if  $result.size() < k$  or  $DTW_{LB} < result[k].DTW$ 
4.       calculate the exact DTW by accessing data file
5.       if  $DTW < result[k].DTW$ 
6.         Result.insert(S,  $DTW(Q, S)$ );
7.       endif
8.     endif
9.   endforeach

```

**Fig. 7.** K-NN algorithm to find out k nearest neighbors of a query time series Q.

Having introduced a lower bound of DTW, we are now ready to introduce the K-nearest neighbor search algorithm. The algorithm is outlined in Figure 8. The whole index file is scanned once (line 1). For every time series  $S$ , its  $DTW_{LB}(Q, S)$  is computed. If we have not yet got  $K$  candidates or the time series  $S$  has smaller lower bound than the  $K^{th}$  objects distance, then we know that the current object *might* be one of the  $K$  closest items. So we must access the data file on disk to get exact time warping distance  $DTW(Q, S)$  (line 3-4). If the exact distance is smaller than the current best-so-far, we add  $S$  to the answer set when we get a better answer (line 5, 6). Note that this approach is similar to, and was inspired by the VA-File technique of [20].

## 4 Experimental Evaluation

Indexing DTW by using tree index structures, such as R-tree, can roughly be separated into two steps. (1) We perform dimensionality reduction on the data, and then index reduced domain with a tree index structure. A set of candidates is filtered out by tracing index tree from root. In this step, if the index structure has high pruning power, most nodes are skipped and the number of pages access becomes much less than naïve sequence scanning. (2) The exact distances between candidates and query data are calculated by accessing data file. If the size of candidate set is big, the number of accessing increases. To reduce the size of candidate set, tight lower bound is necessary. However, it involves increasing the segments, i.e. the number of dimensionality becomes high in multidimensional index structure. Because of “curse of dimensionality”, tree index structure loses its effectiveness in high dimensional space. The number of page access in the step (1) increases largely. There is a trade-off in step (1) and (2). Lower bound is a very important criterion to evaluate dimensionality reduction methods. Like [23], we use tightness  $T$  of lower bound to evaluate the fidelity of representations.

$$T = \frac{\text{Lower Bound of DWT on reduced domain}}{\text{True DTW}}$$

Figure 8 shows the lower bound tightness of Keogh\_PAA [13], Zhu\_PAA [23], LB\_Keogh and LB\_Grid. The LB\_Grid and the original LB\_Keogh do not depend on the number of segments, so they are constant in subfigure (A) and (B). While LB\_Keogh is calculated in original space, LB\_Grid is based on reduced data. LB\_Grid gets tighter lower bound than all other techniques with a very compact representation. It is the key contribution of the paper. On the other hand, as mentioned in Section 2, with the increase of the number of segments for 8 to 16, the tightness of Keogh\_PAA and Zhu\_PAA becomes better. However, by definition, they cannot beat the original Keogh lower bound.

We used two time series datasets to show the performance of our technique compared to other indexing methods for dynamic time warping. The first dataset contains 50,000 random walk data time series. 10-NN queries are tested. Queries are selected randomly from the datasets. Figure 10 shows (A) the number of page accesses which is averaged over 100 experiments. We assume the page size is 8KB. Keogh and

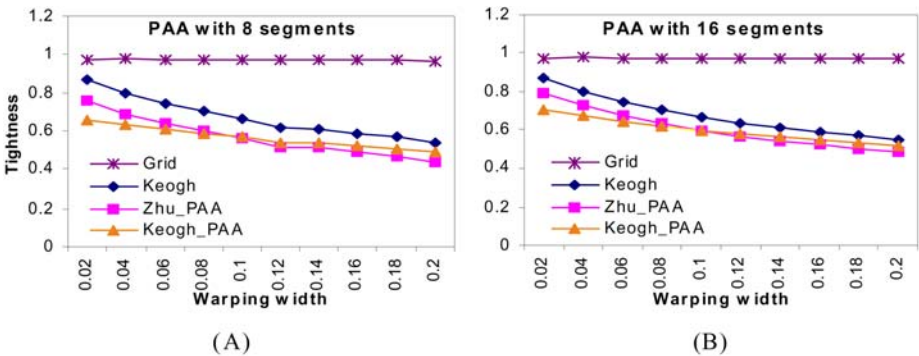


Fig. 8. Lower bound of DWT.

Zhu methods are tested in an R-tree index structure with 8-dimensional index space. We can observe that the number of page accesses of Grid method is greatly lower than other 2 index structures. When the warping width increases, the lower bounds of Keogh\_PAA and Zhu\_PAA becomes very loose and the tree index structures essentially degenerate to sequential scan. Figure 10 (B) shows another dataset called “power data” whose size is 35,000. It was tested in the same conditions of random walk dataset: 10-NN query. Once again the results we report are averaged 100 experiments.

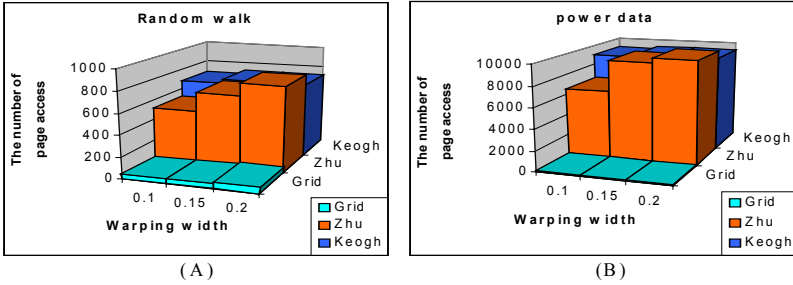


Fig. 9. Comparisons of the number of page access for random walk database.

Our Grid technique uses bit-level data to record omitted information; so it will use more CPU time to reconstruct this information. However compared to the time for accessing hard disk, the extra CPU time is inconsequential. Our work was implemented with C++ in Linux platform. In Pentium 4 PC with 512MB memory.

## 5 Conclusions

In this paper, we introduced a similarity search method for large time series databases using the dynamic time warping distance. We proposed a locally adaptive Grid dimensionality reduction method that can very accurately reconstruct the time series, and thus get very tight lower bound. Almost all competing dimensionality reduction methods for time series [14, 21, 13], have the number of segments (or the dimension of target space) is given as a parameter. In contrast, our Grid technique uses different numbers of segments to minimize the reconstruction errors. That is, it will keep more segments, if the time series can not be represented well with fewer segments. This avoids a loose lower bound for the time series that is difficult to compress. Secondly we proposed an index structure based on filtering and refinement structure.

We have now shown that the Grid representation is at least competitive for Euclidean searching [3], and superior for DTW indexing. In the future we intend to consider the utility of our approach for more complex time series data mining tasks, including anomaly detection [12] and motif discovery [7]. Furthermore, while we have thus far confined our work to one-dimensional time series for clarity of exposition, we strongly believe our work can be generalized to multidimensional time series [19], and offer similar advantages in that domain.

## Acknowledgments

The work in this paper was partially supported by Grant DP0344488 from the Australian Research Council.

## References

1. Agrawal, R., Faloutsos, C., and Swami, A. N.: Efficient similarity search in sequence databases. Proceeding of 4th conference on FODO (1993) 69–84.
2. An, J., Chen, H., Furuse, K., Ishikawa, M., and Ohbo, N.: The convex polyhedra technique: An index structure for high-dimensional space. Proc. of the 13th Australasian Database Conference (2002) 33–40.
3. An, J., Chen, H., Furuse, K., Ohbo, N. and Keogh, E.: Grid-Based Indexing for Large Time Series Databases. Intelligent Data Engineering and Automated Learning, 4th International Conference (2003) 614–621.
4. Beyer, K. S. Goldstein J. Ramakrishnan R. and Shaft U.: When Is "Nearest Neighbor" Meaningful. In proceeding s of 7<sup>th</sup> international conference on database theory (ICDT) (1999) 217–235
5. Beckmann, N., Kriegel, P. H. Schneider, R., and Seeger, B.: The R\*-tree: an efficient and robust access method for points and rectangles. Proceedings of the 1990 ACM SIGMOD International Conference on Management of Data (1990) 322–331.
6. Chakrabarti, K. and Mehrotra, S.: Locally dimensionality reduction: A new approach to indexing high dimensional spaces. Proceedings of 26th International Conference on Very Large Data Bases (2000) 151–162.
7. Chiu, B. Keogh, E., and Lonardi, S.: Probabilistic Discovery of Time Series Motifs. In the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. August 24 - 27, 2003. Washington, DC, USA. (2003) pp 493–498.
8. Faloutsos, C., Ranganathan, M. and Manolopoulos, Y.: Fast subsequence matching in time series databases. Proceedings of the 1994 ACM SIGMOD International Conference on Management of Data (1994) 419–429.
9. Guttman, A.: R-tree: a dynamic index structure for spatial searching. Proceedings of the 1984 ACM SIGMOD International Conference on Management of Data (1984) 47–57.
10. Hale, J. C. and H. L. Sellars. Compression of chemical process data by functional approximation and feature extraction. AIChE J. 42(2), 477. (1981)
11. Katayama, N. and Satoh, S.: The SR-tree: An index structure for high-dimensional nearest neighbour queries. Proceedings of the 1997 ACM SIGMOD International Conference on Management of Data (1997) 369–380.
12. Keogh, E., Lonardi, S and Chiu, W.: Finding Surprising Patterns in a Time Series Database In Linear Time and Space. In the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. July 23 - 26, 2002. Edmonton, Alberta, Canada. (2002) pp 550–556.
13. Keogh, E.: Exact Indexing of Dynamic Time Warping. Proceedings of 28th International Conference on Very Large Data Bases (2002) 406–417.
14. Keogh, E., Chakrabarti, K. Mehrotra, S. and Pazzani M. J.: Locally adaptive dimensionality reduction for indexing large time series databases. Proceedings of the 2001 ACM SIGMOD International Conference on Management of Data (2001) 151–162.
15. Keogh, E. and Foliás, T.: The UCR Time Series Data Mining Archive. [<http://www.cs.ucr.edu/~eamonn/TSDMA/index.html>]. Riverside CA. University of California - Computer Science & Engineering Department (2002).

16. Keogh, E. and Pazzani M. J.: A simple dimensionality reduction technique for fast similarity search in large time series databases. Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD'00) (2000) 122-133.
17. Moody, G.: Mit-bih database distribution [<http://ecg.mit.edu/index.html>]. Cambridge, MA (2000).
18. Seidl, T. and Kriegel, H. P.: Optimal multi-step k-nearest neighbour queries. Proceedings of the 1997 ACM SIGMOD International Conference on Management of Data (1998) 154-165.
19. Vlachos, M., Hadjieleftheriou, M., Gunopulos, D. and Keogh. E.: Indexing Multi-Dimensional Time-Series with Support for Multiple Distance Measures. In the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. August 24 - 27, 2003. Washington, DC, USA. (2003). pp 216-225.
20. Weber, R. Schek, J. H. and Blott, S.: A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces. Proceedings of 24th International Conference on Very Large Data Bases (1998) 194-205.
21. Yi, B. K. and Faloutsos, C.: Fast time sequence indexing for arbitrary lp norms. Proceedings of 26th International Conference on Very Large Data Bases (2000) 385-394.
22. Yi, B. K., Jagadish, H. V. and Faloutsos, C.: Efficient retrieval of similar time sequences under time warping. ICDE2000 201-208.
23. Zhu, Y. and Shasha, D.: Warping Indexes with Envelope Transforms for Query by Humming. Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data (2003) 181-192.



# Fail-Stop Authentication Protocol for Digital Rights Management in Adaptive Information Retrieval System

Zhaofeng Ma and Boqin Feng

Department of Computer Science and Technology,  
Xi'an Jiaotong University, Xi'an, 710049, China  
supermzf@mail.china.com

**Abstract.** Digital Rights Management (DRM) is one of the most important issues in digital library, however it is difficult to balance security and efficiency of DRM system in open network environment, especially when the environment is unauthentic. In this paper, a new and Secure Authentication protocol for Digital Rights Management (SADIRM) is proposed to provide perfect usage control and fair transaction in adaptive information retrieval system, in which a License Finite States Machine (LFSM) is adopted to manage the digital rights under a third part Certificate Authority (CA), thus it is secure and fair for atomic authorization and forced revocation of license dynamically. During the interaction of all transaction stages, a fail-stop digital signature is used for identity authentication, integrity verification, and undeniability. Analysis manifests the proposed protocol is feasible, secure with high integrity and it provides a new and reliable approach for DRM.

## 1 Introduction

Copyright, a legal discipline concerned with the protection of the moral and economic rights of the creators of literacy, scientific and artistic works, is recognized in the Universal Declaration of Human Rights as are the rights to information and culture against outright coping. According to WIPO (World Intellectual Property Organization), Digital Copyright protection consists of two kinds of digital goods which are multimedia type (such as photography, cinema, radio, television, cable, and satellite broadcasting) and software application type together[1-2]. Digital watermark, secure container, Cryptolope, DigiBox et al. are the most famous technologies for multimedia copyright protection that have been studied much more[3-7]. Although software copyright was concerned for a long time, still many things left unresolved which led to illegally copying and usage of kinds of software. Software copyright today is faced with the difficulty of maintaining the delicate balance among the legitimate interests of authors[8-11]. How to protect software from being copied illegally is one of the most important topics for software development and research, upon which special issues in international conferences of Digital Rights Management (DRM'01, DRM'02, DRM'03) about software copyrights were built up to study the commerce copyright legislation and protection.

Unfortunately, though EULA declares more limitations and restrictions, yet it does not provide a valid mechanism and approach for copyright protection thus it can't protect software been illegally copied and used. Hardware-based and pure-software-based protection system are the most common methods for goal software copyrights, among which hardware-based approaches include floppy encryption type, logical circuit encryption type and secure ROM type, while software-based copyright protection approaches consist of product key type, serial No type, and software license type ones as well. However these technologies are difficult to balance security and efficiency of DRM system in open network environment, especially when the environment is unauthentic[10-11].

In this paper, a new and Secure Authentication protocol for Digital Rights Management (SADIRM) based on dynamic license is proposed for trusted rights management in open network environment, where a third part credible Certificate Authority CA[12] is used to manage software license dynamically according to software ID (SID) and user's hardware ID (HID), thus valid user can obtain one and only one license to register the software he bought from its provider. When user needs to migrate the software, he must first logoff and destroy the software resource and commit the logoff result to CA, then he can request the new license for another computer again. The most importance of SADIRM is that it provides a fairly good mechanism to protect software copyright from being copied and reused without any control, where cryptography is used to provide data security and integrity. Analysis manifests SADIRM is easy and secure for software copyright protection.

## 2 SADIRM: Secure Authentication for Digital Rights Management

- (1) Software Provider (SP) binds each software with a sole software identity (SID), and then pass the SID to the third part credible Authentication Center CA[12] (which includes Authentication Server, License Manager Server and License Library).
- (2) Before using the software the user U must first prepare the SID that he got from SP together with the Hardware Identity (HID) that can identify the current computer solely, Then he commit SID and HID to CA to acquire the License for the software to run on the specific machine. CA decide dynamically whether to publish a valid license or not to U according to SID, HID and the current License Status LS for SID, if the request is valid then CA release one and only one license for the current license L, by which U can register the SID to run on HID successfully.
- (3) If the user needs to migrate the software environment, he must first request the logoff password from CA, and then logoff software resource run on the current machine with identity HID, finally commit the logoff result code to CA. After the whole transactional operations finished, U can request another License L' for the same software SID to run on another machine with identity HID', thus the original software resource can be migrated and run correctly again on HID' machine.

In fact, SADIRM controls the copyright through a strong and valid protocol with “Character Association, Atomic Authorization, Restricted Revocation of software copyright” mechanism, by which no matter how many copies are made, but only one of the copies can get the License for the software to be registered, thus it is useless for illegal coping and spread since there’s no profits from coping. SADIRM solves 3 major problems which includes the atomicity of Authorization, free migration of software and entity security of software itself. SADIRM can satisfy the EUEA agreement which entirely solves the problem that software is often illegally copied.

### 3 Protocol Description of SADIRM

#### 3.1 Parameter of SADIRM

**Table 1.** Parameters Definition in SADIRM

PARAMETERS IN SADIRM		
<b>(1) Role Parameter</b>		
U: Software User	AS: Authentication Server	
CS: Certificate Server	V: Service Application	DS: DB Server
<b>(2) Entity Parameter</b>		
SID: Software ID from U	HID: Hardware ID from U	
SID': Software ID stored in DS	HID': Hardware ID stored in DS	
<b>(3) Authentication Parameter</b>		
L: Software License	T <sub>start</sub> : StartTime Requesting L	
T <sub>cur</sub> : Current Time-Stamp	Lifetime: Lifetime of L	
P <sub>Lgt</sub> : Password of Logoffing L	R <sub>Lgt</sub> : Authentication Result of Logoffing	
<b>(4) Integrity Parameter</b>		
K <sub>A,B</sub> : Symmetric Key shared by A,B	Ka,Ka': Public and Private Key from a	
E,D: En/Decryption Function	Sig <sub>K</sub> (m): Signature of m by K	
Ver <sub>K</sub> (s): Verification of s by K	Nonce: Random Nonce	
N <sub>0</sub> : Incremental of Nonce		

#### 3.2 Description of SADIRM

SADIRM includes two sub-protocols which are SRP (software Registering Protocol) and SMP (Software Migration Protocol). Considering justice and security, the following hypotheses are given:

- (1) The third part Authentication Center (CA) is fair, creditable, and is secure in realization.
- (2) Software Provider (SP) is honest that the SID it provides to U is equal to the one that is provides to CA.

##### A. Software Registering Protocol (SRP)

The SRP protocol is for License Request, that is, U gets License from AS, which includes 5 steps. The following is the description of the SRP protocol:

### Stage I: Requesting for License

**Step1:** U commits request vector to AS:  $U \rightarrow AS: R' = E_{K_{U,AS}}(SID \parallel HID \parallel \text{Nonce})$ .

**Step2:** AS authenticates the information from U:  $AS: D_{K_{U,AS}}(R')$ , if  $SID \neq SID'$ , deny the request and return, else goto step3.

**Step3:** After passing the authentication, AS commit the request vector  $R = [SID \parallel HID \parallel \text{Nonce}]$  to CS, CS then decides whether to release the License or not according to decision function  $f_{LRD}$ :

- (1) if  $S_{Lic} = \text{UNRELEASED}$  (here  $HID = \text{NULL}$ ), then CS creates L and  $K_{U,V}$  dynamically, and then signs and encrypts and then sends them to U, finally writes the above information to DS:
  - ① CS: Creates L,  $L = E_{K_V}(SID \parallel HID \parallel P_{run} \parallel T_{start} \parallel \text{Lifetime})$ , where  $P_{run} = S_E \parallel R$ ,  $S_E$  is the signencryption with Version, R is the normal vector for program V to run;
  - ②  $AS \rightarrow U: L' = E_{K_{U,AS}}(\text{Nonce}' \parallel L \parallel K_{U,V} \parallel \text{Sig}_{K_U'}(L))$ , where  $\text{Nonce}' = \text{Nonce} + N_0$ ;
  - ③ DS: Updates license status  $S_{Lic} = \text{RELEASED}$ , and Writes  $T_{start}$ , *Lifetime* and  $P_{run}$  into DS.
  - ④ U receives L from AS, he firstly verifies the validation of Nonce, if  $\text{Nonce}' - N_0 = \text{Nonce}$  is false, then denies to accept, otherwise verifies the Signature of L, if  $Ver_{K_U'}(\text{Sig}_{K_U'}(L)) = \text{true}$ , then U accepts L.
- (2) if  $S_{Lic} = \text{RELEASED}$  or *WAITING*, then it manifests the license for SID has already been released or the case that it has been loggedoff but waiting for the loggedoff result code. Then CS verifies HID:
  - ① if  $HID = HID'$ , then checks whether  $T_{cur} - T_{start} > \text{Lifetime}$  is true, if not, refuses to release L for U, otherwise releases L to U according to (1).
  - ② if  $HID \neq HID'$ , refuses to release license for the request, which is viewed as multiple copies of one software running on different machines or the same software running on different machines.

### Stage II: Registering the Software

**Step4:** U registers V in the lifetime by  $K_{U,V}$  to gets services from V.

- ①  $U \rightarrow V: U$  commits L to V, V decrypt L by  $K_V: D_{K_V}(L) = D_{K_V}(E_{K_V}(SID \parallel HID \parallel P_{un} \parallel T_{start} \parallel \text{Lifetime}))$ , then commit  $T_{start} \parallel \text{Lifetime}$  to CS, CS then computes to validate whether it is true, if true CS passes a password for U to register.
- ② U commits authentication code to V:  $\text{Authen}_u = E_{K_{u,v}}(HID_C \parallel SID_C)$ , where  $SID_C$  and  $HID_C$  is the current SID and HID that to run the software.

**Step5:** V authenticates the validation of  $\text{Authen}_u$  from U according to the L that gets from U.

- ③  $V: D_{K_v}(L) = D_{K_v}(E_{K_v}(SID \parallel HID \parallel P_{run} \parallel T_{start} \parallel Lifetime)), D_{K_{u,v}}(Authen_u) = D_{K_{u,v}}(E_{K_{u,v}}(HID_C \parallel SID_C));$
- ④ if  $HID = HID_C$  and  $SID = SID_C$ , then goto ⑤, else refuses to register;
- ⑤  $V$  allow  $U$  to finish the registration and open services to  $U$ , then abolishes  $L$  and logging the transactional procedure in cipher mode.

## B. Software Transfer Protocol

It is important to ensure the validity of logging request to prevent goal software  $V$  been logged without any control, thus the following procedures is necessary:  $U$  firstly requests logoff password  $P_{lgf}$ , then secondly logoffs the already used software resource, finally commits the logoff result to  $SD$ , thus  $U$  can request License again for another computer when necessary. Software Migration Protocol (SMP) includes two sub-protocols which are requesting logging password stage and logging software resource stage. The following is the description of the SMP protocol:

Considering possible attack and treating, the software migration protocol must satisfy the conditions:

- (1) The interval of re-requesting the license must expire the lifetime that  $CS$  declared in  $L$ , thus so  $U$  can't personate to request to different valid licenses that fits to run the software on different machine.
- (2) To avoid software being logged without any control, when valid user wants to logoff the software, he must first get logoff password and do the logoff operation.
- (3) During the logoff procedure, all operation must be done in transaction mode which meet the demand of ACID properties. If and only if  $CS$  confirms that the logoff is finished then  $U$  can request another License again to get its service.

The SMP protocol is divided into 2 stages which include 5 steps, the following is the description of SMP:

### Stage I: Requesting for Logoff Password

**Step1:**  $U \rightarrow AS: R' = E_{K_{U,AS}}(SID \parallel HID \parallel Nonce);$

**Step2:** As corresponding operation in SRP, if the  $R'$  can't pass the authentication, then deny to do anything for  $U$ , otherwise, goto step3;

**Step3:**  $CS$  checks the license status  $S_{Lic}$ , and according to which decides whether to release logoff password  $P_{lgf}$  or not:

- (1) If  $S_{Lic} = RELEASED$  or  $S_{Lic} = WAITING$  and  $HID = HID'$ , then authentication whether the request lifetime is expired or not, if  $(T_{cur} - T_{start} > Lifetime)$  is false, it manifests that the pre-released license is still in its lifetime, then deny the request. otherwise, it means that the pre-released license is expired its lifetime, then  $AS$  creates logoff password  $P_{lgf}$  and pass it to  $U$ , and update critical Data in  $DS$ .
- ①  $AS$  creates  $P_{lgf}: P_{lgf} = E_{K_v}(SID \parallel HID \parallel P_{Stop} \parallel W_{BCK})$ , where  $P_{Stop} = S_E \parallel P$ ,  $S_E$  is the signencryption with version, and  $P_{Stop}$  is the Stop Vector to pre-

vent  $V$  to run, while  $W_{BCK}$  is the Authentication Word that need to return to AS so that AS can authentication whether the logoff transaction is finished or not.

- ② AS Updates  $S_{Lic}=WAITING$ , and set  $Nonce':=Nonce+N_0$ , then sign and encrypt the  $P_{lgf}$  to U:  $C'=E_{K_{U,AS}}(Nonce' \parallel P_{lgf} \parallel K_{U,V} \parallel Sig_{Ku'}(P_{lgf}))$ .
- ③ U gets  $P_{lgf}$  from AS and verifies  $Nonce'-N_0=Nonce$  is true or not. Is true, then verifies Signature of  $P_{lgf}$  if  $(Ver_{K_V}(Sig_{K_V'}(P_{lgf})))=true$ , then accepts, otherwise refuses
- (2) if  $S_{Lic}=UNRELEASED$  or  $S_{Lic}=RELEASED$  but  $HID \neq HID'$ , the deny to release the logoff password to U.

### Stage II : Logoffing Software Resource

U logoffs the software resources by the request logoff password  $P_{lgf}$ , then commits logoff result authentication word  $W_{BCK}$  to AS, then if the  $W_{BCK}$  is verified to true, he can then request another License to register and use the software again.

**Step4:** U decrypts the  $P_{lgf}$  from AS, then verifies the signature, if the signature is true, then do the following operations:

- ①  $U \rightarrow V: P_{lgf} \parallel Authen_u$ , where  $Authen_u = E_{K_{u,v}}(HID_C \parallel SID_C)$ ;
- ②  $V: D_{K_V}(P_{lgf}) = D_{K_V}(E_{K_V}(SID \parallel HID \parallel P_{Stop} \parallel W_{BCK}))$ ,  
 $D_{K_{u,v}}(Authen_u) = D_{K_{u,v}}(E_{K_{u,v}}(HID_C \parallel SID_C))$ ;
- ③ if  $HID=HID_C$  and  $SID=SID_C$  then goto step5, otherwise denies to logoff the software resources;

### Step5:

- ① U logoffs the software SID that runs on the current HID machine, and then destroys resources of SID on HID, and locks the whole logoff operation in transaction mode, finally creates the logoff result  $R$ ,  $R=Logoff(SID \parallel HID \parallel P_{lgf})$ , where  $R = bLogoffed \parallel R_{Lgf}$ . If  $bLogoffe=false$ , it manifests the logoff operation failed, then  $R_{Lgf}=NULL$ ; otherwise  $R_{Lgf}=E_{K_V}(SID \parallel HID \parallel W'_{BCK})$ . U returns  $R_{Lgf}$  to AS:  $E_{K_{U,AS}}(SID_C \parallel HID_C \parallel R_{Lgf})$ .
- ② AS decrypts and verifies the validation of  $R_{Lgf}$ , if  $SID_C=SID$ ,  $HID_C=HID$  and  $W'_{BCK}=W_{BCK}$  are all true, then accepts the results and updates  $HID'$  and  $S_{Lic}$ ,  $HID'=NULL$ ,  $S_{Lic}=UNRELEASED$ ; otherwise refuses the logoff results.

### 3.3 Fail-Stop Digital Signature for SADIRM

To enhanced the security and undeniability during the system interaction, we adopted Fail-Stop Digital Signature [13-15] for inter-authentication in SADIRM, which is applied in the above protocols in each stages.

**(1) System Parameters**

$p, q$  are large primes  $p=2q+1$ , which are difficult to solve discrete logarithm in  $Z_p$ ,  $\alpha \in Z_p^*$ , whose order is  $q$ ,  $\alpha_0 \in Z_q^*$ ,  $\beta = \alpha^{\alpha_0} \bmod p$ ,  $p, q, \alpha, \beta, \alpha_0$  are decided by CA.  $p, q, \alpha, \beta$  are public parameters, while  $\alpha_0$  are secret parameter.

The message scope is  $M=Z_q$ , Signature Scope is  $S=Z_q \times Z_q$  and  $K=(r_1, r_2, a_1, a_2, b_1, b_2)$ , where  $a_1, a_2, b_1, b_2 \in Z_p$ , and

$$r_1 = \alpha^{\alpha_1} \beta^{\alpha_2} \bmod p, \quad r_2 = \alpha^{b_1} \beta^{b_2} \bmod p$$

**(2) Signature Procedure**

For Message  $M \in Z_q$  and  $K=(r_1, r_2, a_1, a_2, b_1, b_2)$ , CA compute:

$$y_1 = (a_1 + Mb_1) \bmod q, \quad y_2 = (a_2 + Mb_2) \bmod q$$

then

$$S = \text{Sig}_K(M) = (y_1 \parallel y_2)$$

is the signature of  $M$ .

**(3) Verification Procedure**

If and only if the following expression is true, the signature system is right:

$$\text{Ver}_K(M, S) = \text{true} \Leftrightarrow r_1(r_2)^M \equiv \alpha^{y_1} \beta^{y_2} \bmod p$$

In fact, it is easy to prove that:

$$\begin{aligned} r_1(r_2)^M \alpha^{y_1} \beta^{y_2} &\equiv \alpha^{a_1 + Mb_1} \beta^{a_2 + Mb_2} \bmod p \\ &\equiv (\alpha^{a_1} \beta^{a_2})(\alpha^{b_1} \beta^{b_2})^M \bmod p \equiv r_1(r_2)^M \bmod p \end{aligned}$$

**4 Analysis of SADIRM**

**Theorem 1.** If  $SID$  and  $HID$  that  $U$  commits are valid, then through  $SRP$ ,  $U$  can securely get one and only one valid license  $L$  to register the software and get its services.

**Theorem 2.** According to  $SMP$ , Legal user  $U$  can get valid logoff password  $P_{Lgf}$ , with which  $U$  can logoff and transfer the software resource to another machine.

**Proof (Theorem 1).** According to  $SRP$ , when  $U$  commits  $\text{Req} = E_{K_{U,AS}}(SID \parallel HID \parallel \text{Nonce})$ , by  $f_{LRD}$ , if and only if  $(SID = SID' \wedge HID = NULL \wedge S = UNRELEASED)$  or  $((SID = SID' \wedge HID = HID') \wedge (T_{\text{cur}} - T_{\text{start}} > \text{Lifetime}))$  is satisfied,  $AS$  releases License  $L$  to  $U$ . Once the license of  $SID$  is released, when  $SID$  is not logoffed,  $U$  can't get another License to register on another machine. While the License is in lifetime, that is  $T_{\text{cur}} - T_{\text{start}} < \text{Lifetime}$ ,  $U$  can't request the license for  $SID$  to register on another  $HID$ , so replay attack will not succeed. Thus when the request from  $U$  is valid,  $U$  can get one

and only one License L to register and get its services, therefore the SRP protocol is atomic and sole in releasing license.

During each step in SRP, all communication is done in cipher mode, if the cryptography is secure enough, the interaction is under high level protection. U verifies the identity of AS through  $\text{Nonce}' - N_0 = \text{Nonce}$ , according to the assumption that CA is credible, the License L U got from AS is valid, and the L can pass the verification. Although U can personate another L', however he won't know the exact timestamp that stored in DS, when registering the L' will not pass the verification, thus the integrity of L is ensured. Moreover, for AS is credible, then L is true. so through SRP protocol, U can securely get license L which the integrity and undeniability are enhanced, through which U can get its services.

Considering kinds of attacks, if U didn't logoff the License L but to get another license L' to run SID on another machine, however the timestamp is controlled by AS, thus U can't modify  $T_{\text{Start}}$ ,  $T_{\text{Cur}}$  and Lifetime. Thus replay attack won't succeed. While U finishes registering SID and he tries to request another License L', for the license status  $S_{\text{Lic}} = \text{RELEASED}$ , AS will not release License to U. Otherwise he must firstly logoff the previously registered License L to get request new license.

Thus from the above 3 aspects, the release strategy of SRP is atomic, the content of L is undeniable, and registering is controlled strictly by system timestamp, therefore through SRP, U can one and only one License L to get services from SID ■.

## 5 System Implementation of SADIRM

### (1) Software Architecture of SADIRM

Based on SADIRM protocol we have implemented the software completely, which is built on the principle of layers, modularization and expansibility, where Microsoft Visual C++6.0 is used for application GUI and algorithm mplementation, while Microsoft SQL 2000 (Enterprise Edition) is for database support. The SADIRM system consists of 3 different executable applications (SmartLicAgent, AuthenServer, and, SmartLicManager and a dynamic linking library (SmartMonitor.dll, SmartMonitor.lib) for access control and data security, and SmartLicManager interacts with database for data manager, especially the independent cryptography library ICL is for data encryption and decryption in middleware style which enhances the performance of the SADIRM system greatly in extendibility, migration. Fig. 1 is the software architecture of SADIRM.

### (2) Data Encrypt and Decryption

The Independent cryptography library (ICL) is implemented as middleware for data En/Decryption and digital signature, the fail-stop algorithm for digital signature, and DES algorithm for data encryption. Considering convenience and efficiency, the ICL algorithm is complemented in pure C instead of Microsoft CryptoAPI or Java Crypto Class, especially parts of the public key cryptographic algorithm is implemented in ASM language for high speed.



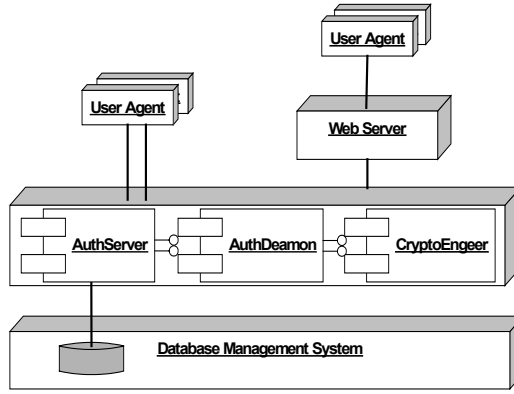


Fig. 1. Software Architecture of SADIRM

```

<?xml version="1.0" ?>
- <CPSec>
- <CPSecEntity>
- <EntityInfo>
  <SoftwareID>{5FB52551-A9A3-4b57-884F-0053C76C3060}</SoftwareID>
  <HardwareID>F2A93343 59E8168C</HardwareID>
</EntityInfo>
- <UserInfo>
  <User>Ma Zhaofeng</User>
  <Company>xjtu</Company>
  <E_Mail>supermzf@mail.china.com</E_Mail>
</UserInfo>
</CPSecEntity>
- <CPSecLicense>
  <SoftValidation>IS_VALID</SoftValidation>
  <SoftwareStatus>RELEASING</SoftwareStatus>
  <CPSecLicense>779ED58E 25773D70 2B817E21 894CE158 64457FC9 9C53AE(
779ED58E 25773D70 2B817E21 894CE158 25C49B72 3D266C9B 26B4C0F6
DDBE070B BD8D27BE 75CF93A3 779ED58E 25773D70 2B817E21 894CE158
9B4A858 7C4F9F4AE 6EDFC1DC 596DD859 4583B9C2 2E5F061 1957ACBB
2B817E21</CPSecLicense>
</CPSecLicense>
</CPSec>

```

Fig. 2. License Format of SADIRM

### (3) Character Data Creation

Character data in SADIRM includes software character SID and hardware character HID. SID is created by the GUID executable application, which has the uniform format, while HID is built on 3 critical characters of computer itself, which are CPU serial NO (C) , Hardisk Serial NO(H),MAC serial NO(M), thus HID is the mapping from C, H and M, that is:

$$HID = F(C, H, M) = f_{hash}(g_{en}(k_{logic}(C, H, M)))$$

where F is the compound mapping from  $f_{hash}$ ,  $g_{en}$  and  $k_{logic}$ , here  $k_{logic}$  the compound logical operation,  $g_{en}$  is the secure encryption function,  $f_{hash}$  is the secure

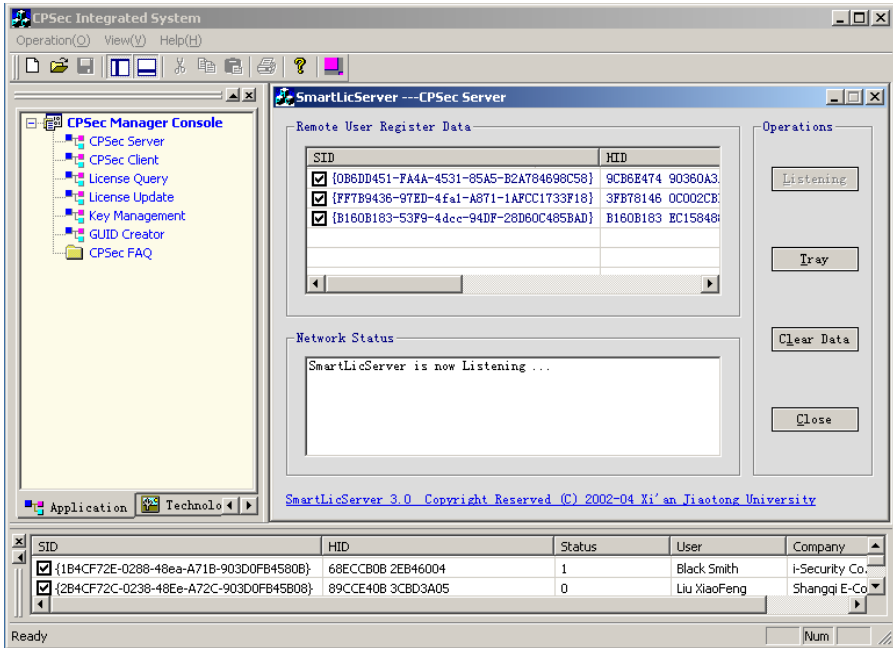


Fig. 3. Integrated GUI of SADIRM Server System

Hash function. Considering that not all computers are equipped with Network Adapter, while only PIII type CPU has serial No., but hardisk serial No. always exists.

#### (4) Security of Software Entity Itself

The most important and critical feature of SADIRM software implementation is that an enhanced signature mechanism is used to protect the software itself from being cracked, where segment signature and random verification mechanism is used to implement the software integrity and security. SADIRM can resist out attack such as SoftICE, but without sensitive information leakage thus SADIRM can ensure software entity itself is secure and strong enough to resist attack or crack. Then when can't obtain MAC serial No, or CPU serial No, default value is used instead. Fig. 2 is the license created by SADIRM server, and Fig.3 is the Integrated GUI of SADIRM Server System.

## 6 Conclusion

Aiming at the current demerits of software copyright protection solutions, a novel and secure copyright protection solution SADIRM based on dynamic license is proposed to solve these problems. SADIRM provides a fairly good mechanism for atomic authorization, complete revocation of software resource to protect software from being copied and reused without any control, where cryptography is used to provide data

security and integrity, and analysis manifests SADIRM is valid, generic, and secure for software copyright protection and meet the demand of EULA in a consistent way. As the application, we have applied the inter-authentication protocol SADIRM for DRM in web-based adaptive information retrieval system that we developed for personalized information push-delivery.

## References

1. Bordoloi B, Ilami P, Mykytyn PP, Mykytyn K.: Copyrighting computer software: The "look and feel" controversy and beyond. *Information & Management*, Vol. 30(5), 1993: 211-221
2. Milagros DC.: Copyright in the digital environment. *International Information and Library Review*. Vol. 29(2),1997: 201-204
3. Su JK, Hartung F, Girod B.: Digital watermarking of text, image, and video documents. *Computers & Graphics*, Vol. 22(6),1998: 687-695
4. Lin PL.: Digital watermarking models for resolving rightful ownership and authenticating legitimate customer. *Journal of Systems and Software*, Vol. 55(3), 2001: 261-271
5. Eskicioglu AM, Delp EJ. An overview of multimedia content protection in consumer electronics devices. *Signal Processing: Image Communication*, Vol. 16(7), 2001: 681-699
6. Waller AO, Jones G, Whitley T, Edwards J, Kaleshi D, Munro A, MacFarlane B, Wood A. Securing the delivery of digital content over the Internet. *Electronics & Communication Engineering Journal*, Vol. 14(5), 2002: 239 -248
7. Lee WB, Chen TH.: A public verifiable copy protection technique for still images. *Journal of Systems and Software*, Vol. 62(3), 2002: 195-204
8. Eskicioglu AM, Delp EJ.: An overview of multimedia content protection in consumer electronics devices. *Signal Processing: Image Communication*, Vol. 16(7), 2001: 681-699
9. Biehl I, Meyer B.: Cryptographic methods for collusion-secure fingerprinting of digital data. *Computers and Electrical Engineering*, Vol. 28(1), 2002: 59-75
10. Oz E.: Acceptable protection of software intellectual property: a survey of software developers and lawyers. *Information & Management*, Vol. 34(3), 1998: 161-173
11. Yoon K. The optimal level of copyright protection. *Information Economics and Policy*, Vol. 14(3), 2002: 327-348
12. Schneier B.: *Applied cryptography-protocols, algorithm and source code in C*. Second Edition, New York: John Wiley & Sons Inc, 1996
13. van Heyst, E., Pedersen, T.P. and Pfitzmann, B.: New constructions of fail-stop signatures and lower bounds, *Advanced in Cryptology:CRYPTO'92, LNCS*,Vol.740(1993):15-30
14. van Heyst, E., Pedersen, T.P.: How to make efficient fail-stop signatures, *Advanced in Cryptology: EUROCRYPT'92, LNCS*, Vol.658 (1993): 363-377
15. Damgard, I. B., Pedersen, T.P.,and Pfitzmann: On the existence of statistically hiding bit commitment schemes and fail-stop signature, *Journal of Cryptology*, Vol.10, 1997:163-194

# Effective Web-Related Resource Security Using Distributed Role Hierarchy\*

Gunhee Lee<sup>1</sup>, Wonil Kim<sup>2</sup>, Dong-kyoo Kim<sup>3</sup>, and Hongjin Yeh<sup>1</sup>

<sup>1</sup> Graduate School of Information Communication, Ajou University, Suwon, Korea  
icezzoco@ajou.ac.kr

<sup>2</sup> College of Electronics and Information Engineering, Sejong University, Seoul, Korea  
Tel: +82-2-3408-3795

wikim@sejong.ajou.ac.kr  
<sup>3</sup> College of Information Technologies, Ajou University, Suwon, Korea  
dkkim@ajou.ac.kr

**Abstract.** The recent trend of access control on the Web-related resource is that the service provider differentiates services according to the user's attributes such as membership class or job position. For proper and simple representation of the relationships between user's attributes and permission, role-base access control is used on the Web. As the size of the organization increases, does the number of membership classes or job positions. Naturally, the role hierarchy becomes too complicated to administrate properly. The lack of proper role administration makes the system vulnerable in access control. Moreover, the web services are distributed and categorized by the type of service. It is difficult that a centralized server controls the distribute environment properly. In this paper, we propose a distributed role hierarchy module that can manage the role hierarchy effectively and practically. The proposed system employs the distributed local role hierarchy, and the combination of them represents a logical global role structure. Distributed role hierarchy modules manage local role hierarchy that is related to each of them.

## 1 Introduction

Today the Web becomes an essential part of everyday activity. In companies, schools, and public offices, people are hooked on computers to connect to the Web. Many organizations, such as company or government, use the web-based work management system. In this environment, most Web-related resources are managed by service provider. The permission of resource usage depends on each person's job position or membership class, thus the service provider should support different levels of services according to the user's attributes.

To satisfy this requirement, many security specialists have researched on access control methods. One of the access control method is role-based access control (RBAC) [1]. RBAC is suitable technology for managing and enforcing security policy in large-scale enterprise-wide systems [2]. Thus RBAC has been

---

\* This study was supported by the Brain Korea 21 Project in 2004.

used on the Web as access control system [3], [4], [5]. These researches manage the role hierarchy with the centralized manner. However many web services are distributed and categorized. Moreover, in the large organization, the role hierarchy is very complex, and only a specified part of role hierarchy is authenticated for proper access [6]. The current access control system provides this complex role hierarchy using only single server. In this centralized method, the role administration is difficult to maintain proper and consistent role hierarchy.

In this paper, we propose a distributed approach to role administration in RBAC for secure Web environment. We propose role hierarchy mechanism with decentralized role hierarchy module that administrates the whole role structure properly and effectively. When a user asks permission for resource, related Web server sends a verifying request to the role hierarchy module. It checks the availability of the proper role for the given request. If the module can not make a proper decision, it sends the request to upper level role hierarchy module. This approach removes the difficulties of role administration since the role hierarchy is simpler than centralized role hierarchy. It is safer than centralized approach since each role hierarchy module can concentrate on security of corresponding role hierarchy. Moreover, proposed method processes the management of role and user's request effectively.

This paper is organized as follows. We describe access control on the Web, RBAC concept, and distributed web security in section 2. In section 3, we explain existing RBAC system on the Web and its disadvantages. In section 4, we describe proposed decentralized approach in detail. This is followed by a case study of the proposed approach in section 5. Section 6 concludes.

## 2 Web Security

### 2.1 Access Control on the Web

The Web access control system consists of three parts such as making the policy, collecting the user and the system data, and enforcing the policy. Policy is represented by language and is formalized by rule, logic, or formula. User and environment data are status information such as referred page, connection time, and requested action. Enforcing policy is based on the value of these attributes. Bauer et al. proposed the logic based access control on the Web [7]. Policy based access control system was introduced by Ungureanu et al [8]. Emin Gun Sirer et al. proposed the access control system that applying rules depending on the users' attributes [9].

Generally, the Web access control policy defines permission for individual user and resource. However, the security of large-scale organization focuses on higher-level policies that are derived from laws, ethics, regulations, or generally accepted practices. The policy of such organization usually requires not only an ability to control access to information but also an ability to control the actions of individuals [2]. In that environment, the relations among users, resources, and cases are too complicated to be managed manually. Therefore, it is very hard to achieve access control consistency for this kind of security system. For example,

professor should be able to gain full control of the courses that he/she teaches. While he/she should not gain control of other courses, he/she may read them. In this case, the traditional method has to define many rules for every case. This is a formidable and difficult task.

## 2.2 Role-Based Access Control

In 1992, David Ferraiolo and Richard Kuhn introduced RBAC (Role-based access control) [1]. RBAC can be a good solution to the problem that we have described in the previous section. In RBAC model, a user has multiple roles and a role has multiple permissions. Permission is represented by the relation of resource and operation. Hence the policy in RBAC is always concerned with what the user's role is [10]. It is much more efficient and time-saving than traditional access control methods.

Conflicts of interest in a role-based system may arise as a result of a user's gaining authorization for permissions associated with conflicting roles [11]. For example, if one role requests expenditures and the other role approves them, the system must prohibit the same user from being assigned or active to both roles.

The number of roles can be in the hundreds or thousands, and users can be more than that in large organizations. Managing these roles, users, and their relations is a formidable task [6]. To solve this problem, administration of RBAC uses RBAC itself. Administration of RBAC is important, and must be carefully controlled to ensure that policy does not drift away from its original objectives.

## 2.3 Web Security Using RBAC in Distributed Domain

Freudenthal et al. introduced *dRBAC*, which supports coalition among different organizations [5]. *dRBAC* is a decentralized trust management and access control mechanism for systems that span multiple administrative domain. This system uses PKI to establish trust between deferent organizations. *dRBAC* uses the role to define controlled activations. To use another organizations service, *dRBAC* employs role delegation concept. This system allows the third-party delegation. By referring directly to the role originator's namespace, an authorized entity delegates roles created by another entity. The valued attributes are also used in order to support varying levels of access for the same resource.

Bacon et al. have developed *OASIS*. This is a RBAC architecture for achieving secure interoperation of independently managed services in an open, distributed environment [12]. Each service is responsible for classifying its users into named roles. A user is authenticated by presenting his/her credential to a service, after which the service issues a role membership certificate (RMC). The RMC includes the identifying aspects of user. A user is authorized by specifying the method that he/she may invoke. For the authorization, privileges are expressed by the formal language based on Horn clauses, and the policy includes constraints that must be checked.

These two RBAC systems for distributed environments mainly addressed the solution of inter-domain problems. Each independent domain uses a role

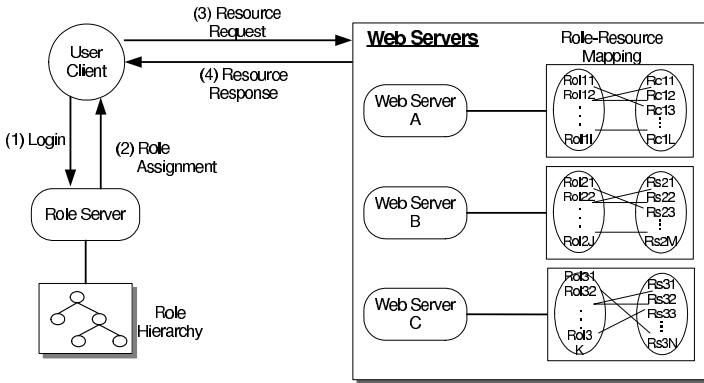
management server. Therefore, for a domain, they have the same problems as mentioned in section 3.2.

Tari and Chan released *I-RBAC*, role-based access control for intranet security [13]. Every server such as mail, file, and database server has its own role hierarchy called local role hierarchy. For the globally used network object, global role hierarchy is used. Global roles take individual local roles into account; thus, a global role can have multiple local roles in multiple servers. Each server has local role database and a global role database manage all the global roles. In this system, the role structure is too complicated to maintain the consistency between global and local roles. Because the intranet is an endless changing environment, the alteration of roles must be managed to ensure consistent databases.

### 3 Current RBAC on the Web

#### 3.1 Current RBAC Mechanism on the Web

Currently general RBAC systems use a centralized role server in order to assign a role to a user in the Web environment. Before a user requests a resource from the Web server, he or she must acquire the role from the role server. At this time, the role server issues a certificate to the user. User's attributes and role are in the certificate. After connecting to the Web server, the user client sends user's attributes such as an identity to the Web server. Then the server verifies user's attributes and decides user's access. Fig. 1 describes the schematics of this process.



**Fig. 1.** A schematic of existing system.

The role server has a global role hierarchy with constraints. It assigns a role to a user by certificate, and the Web server assigns the corresponding permissions to the role in the certificate. Some implementations employ intermediate account to assign the permission to role [14], [15]. The role hierarchy and the constraints are very complex and modified manually by the security administrator.

### 3.2 Problems of RBAC on the Web

In the existing centralized approach, whenever a user requests a service or a resource, the user has to obtain proper credential such as cookie or certificate from the role server. Since all the transactions are centralized to the role server, this approach causes the vulnerability of security such as resource consumable attack. If the role server is attacked and finally malfunctioned, the entire system has to be shut down.

Moreover, the complexity of the role hierarchy in the large-scale system is too high to administrate it properly with a centralized role server. When the complexity of the role hierarchy is very high, the number of role and related permissions can be more than thousands. It is not secure and effective for a centralized role server to administrate this large scale system.

In the practical environment, the whole permission-role relation in the hierarchy is related to many distributed web servers. These distributed web servers provide different services, and these services are categorized by the level of security and workflow of the service. Thus existing approach is not suitable to practical environments. In the following chapter, we propose a new system that overcomes this problem and ensures proper and secure access to the Web resources.

## 4 Proposed RBAC System on the Web

To cope with the problem described in the previous section, we propose a system that employs distributed role hierarchy. In the proposed system, the global role hierarchy is divided into several local role hierarchies according to the services of each web server. Each local web server has own role hierarchy module. Every role hierarchy module manages own local role hierarchy. A user can initiate a service request to a local web server that has a lowest level role hierarchy module. When the web server cannot provide requested service due to limitation of role, it sends the request to the higher-level web server to provide proper permission for the requested service.

### 4.1 Role Group

Generally, the locality of the service comes from the relevance and the level of the service. In the large organization, a set of related services is provided by a web server. Any other services are also categorized into related groups that are distributed to web servers. In order to achieve this, service provider considers different levels of services. Therefore, security administrator creates roles according to the service level and the role hierarchy is designed according to this group. According to this process, the role group is obtained, after which it is used to organize local role hierarchy.



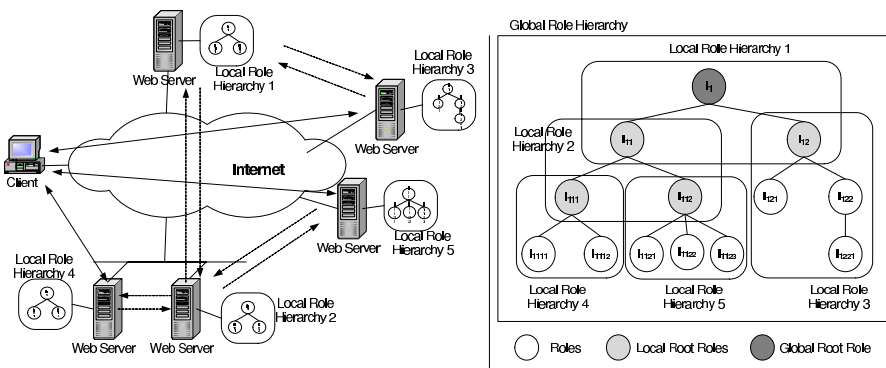
## 4.2 Distributed Role Hierarchy

In the proposed system, we employ two kinds of role hierarchy; global role hierarchy (GRH) and local role hierarchy (LRH). GRH is a logical role hierarchy that describes the whole relations among all the roles in the organization, while LRH is a physical role hierarchy that describes the relations among the roles in the categorized role group. In other words, GRH is a schematic role hierarchy of the entire system and LRH is practically divided into small groups depending on the role group. The right side of Fig. 2 shows the concept of these role hierarchies. Each LRH is managed by the local role hierarchy module in the web server. The level of the service decides the hierarchy structure of role hierarchy module. Each LRH has the local root role that is a bridge to connect with higher level.

## 4.3 Explanation of the Proposed Model

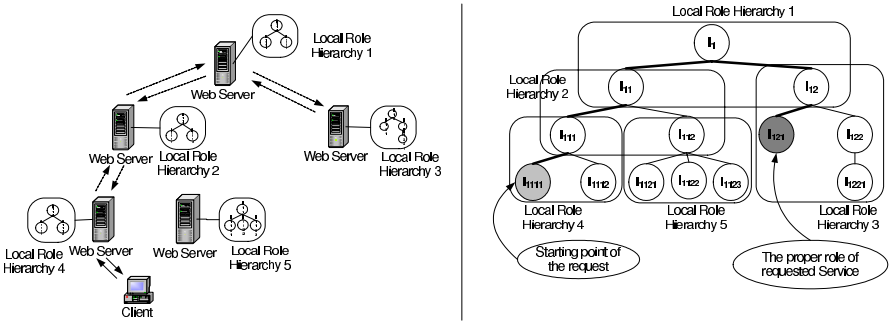
Fig. 2 shows the schematic diagram of the proposed system. Several distributed web servers provide different levels of services. This is a general case in a large organization. For example, the main office may be in New York City and branch offices are in Los Angeles, San Francisco, and Boston. While the headquarter manages the whole process of the company, such as making product, and marketing, the branch offices conduct the support service for the customer and distribute the product. The whole organization operates all together, and this whole role hierarchy (global role hierarchy for this company) may be divided into several local role hierarchies by the related branch.

A user requests a service by connecting to web server. A user's request is sent to a lowest local web server. Role hierarchy module in the given web server checks whether requested service should be handled or not. Handling of the



**Fig. 2.** The schematic diagram of proposed system; a solid line means a user's request, while a dotted line means a traverse of role hierarchy to find available web server. A global role hierarchy is divided into 4 local role hierarchies depending on the role group.

request means that the role hierarchy module can assign the proper role to the user except for the local root role. If it can handle the request, appropriate services are granted. Otherwise, the role hierarchy module requests the services to another higher role hierarchy module including own local root role. With the same manner in the lower role hierarchy case, the higher role hierarchy module checks the availability of handling of the user's request.



**Fig. 3.** An example of the reverse traverse: Even the root role can not handle the request, it traverses down to the alternative route in the global hierarchy.

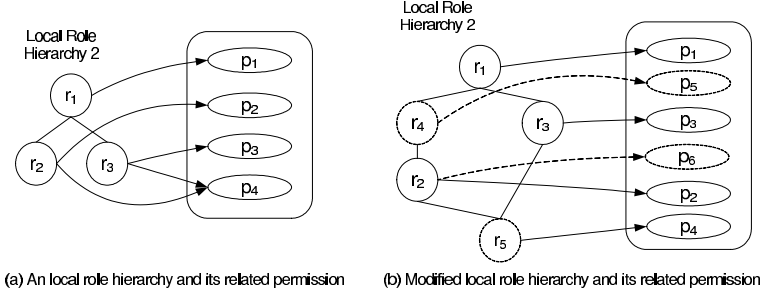
In some cases, the user's request can be handled by another equivalent level role hierarchy module, not the higher level module. For this case, we use reverse traverse method. The requested services can be provided by reverse traverse from a root. When the user's requests are not handled even though the highest role hierarchy module checks the ability of handling it, the proposed system traverses again through the other way of the global role hierarchy. Fig. 3 shows this case.

#### 4.4 Management of Role Hierarchy

In the role hierarchy, creation/deletion of role and assign/revoke of permission to role are often occurred depending on the situation. As the size of the organization larger, these changes occur frequently. The proposed distributed method manages this situation effectively.

In the distributed approach, the management of the change is not drastically different from the centralized approach. All the services are categorized when the system is created and these categories form a hierarchy with level. Therefore, except for the local root role, all the changes are handled locally. The change of the local root role (such as  $l_{111}$  local root role of LRH 4 in the Fig. 3) is occurred in the higher module (local role hierarchy 2 in our example). After changing, the result is propagated to the lower module (local role hierarchy 4 in our example). Fig. 4 shows an example of a local role changing procedure.

In the Fig. 4, there are 3 roles, such as  $r_1$ ,  $r_2$ , and  $r_3$ , and 4 permissions, such as  $p_1$ ,  $p_2$ ,  $p_3$ , and  $p_4$ . The relations of role and permission are as follows;  $r_2$  has



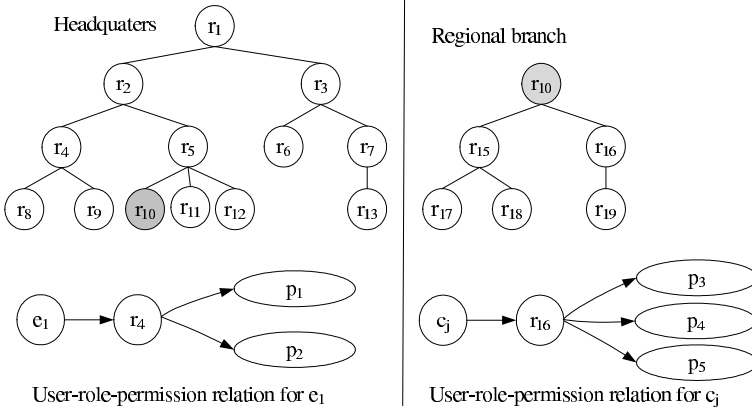
**Fig. 4.** An example of modifying roles: The change of the  $r_2$  (e.g. local leaf role of LRH 2 and local root role of LRH 4) propagate to the related lower level role hierarchy module.

$p_2$  and  $p_4$ ,  $r_3$  has  $p_3$  and  $p_4$ , and  $r_1$  has  $p_1$  and inherit  $p_2$ ,  $p_3$ , and  $p_4$  from  $r_2$  and  $r_3$ . First, the new permission  $p_5$  is inserted, and the new role  $r_4$  is created and is inserted to the hierarchy for the  $p_5$ . The new permission  $p_6$  is added to the system and is assigned to  $r_2$ . Last, to satisfy the least privilege condition, the new role  $r_5$  is created and permission  $p_4$  is assigned to it.  $r_2$  and  $r_3$  have the permission  $p_4$  through the inheritance. These set of changes is related to the local role hierarchy except for the case of  $r_2$ . Changing of the  $r_2$  must be propagated to the related lower role hierarchy module. To do so, current module that has the modified role  $r_2$  sends the information about the  $r_2$ , such as set of permissions and constraints, to the lower level module.

## 5 Case Studies of the Proposed System

We use a network of world-wide travel agent consisting of three levels of branches in the case study. The top-level branch is for the headquarters. The second levels (regional level) are for the branches related to wide regions, and the last levels (sub-regional level) are for the branches related to small regions. Each branch conducts business such as marketing travel package, reservation of accommodating and airplane, and agency work for a visa and passport. The sub-regional level web server provides reservation service related to traffic and accommodation. The regional level web server provides visa and passport agency service. An employee information and job information are provided by a server to which he/she belongs. The result of the business is reported to the headquarters, by which it is appraised. The headquarters makes a decision on the financial affair and the existence of a branch. We explain two practical examples; the use case of employee and the use case of customer. The role hierarchy and user-role-permission relation are shown in Fig. 5.

**Case 1.** Employee  $e_i$  is the executive officer belonged to headquarter, and  $e_i$  has a role  $r_4$  that has two permission  $p_1$  and  $p_2$ . A permission  $p_1$  means reviewing the financial report and  $p_2$  is the appraising of the report. Suppose that  $e_i$  is on



**Fig. 5.** Role hierarchies and user-role-permission relation for two case study.

a business trip to visit Los Angeles branch in order to appraise the branch. To do it,  $e_i$  need to use  $r_4$  that has  $p_1$  and  $p_2$ . Employee  $e_i$  requests permission  $p_1$  to review financial report of Los Angeles branch. In this case, the regional level web server takes the request, but the server does not handle  $e_i$ 's request since related role hierarchy module does not have the role  $r_4$ . Thus the module sends the request to the higher level role hierarchy module. Higher level module finds the role  $r_4$  and checks the relation to verify that  $e_i$  can use the role. At last, the employee  $e_i$  can review the report and appraise the branch. This appraisal report is recorded to the database through the Web and is reviewed by the CEO.

**Case 2.** When a customer  $c_j$  residing in sub-region wants to get a visa for U.S., he/she sends the request to the visa agency. To do so, he/she connects to the homepage of the travel agency, and then  $c_j$  complete the request form. He/she can change any time before the acceptance of the request and can check the process of the treatment of request via homepage. To do so, the role  $r_{16}$  is needed in the regional branch. The role  $r_{16}$  has three permissions such as  $p_3$ ,  $p_4$ , and  $p_5$ . Each of permission means that filling/reviewing the form and checking the process in sequence. When a sub-regional level web server where he/she lives takes the  $c_j$ 's request, the role hierarchy module checks the availability of handling the request. However, because visa agency service is the role of regional level web server, the module sends the request to higher-level module. Since the higher level module has the proper role  $r_{16}$ , and check that  $c_j$  may use this role. Since the user-role relation shows that he/she can use it,  $c_j$  will get the visa.

## 6 Conclusions

In the most of the Web environments, web servers are distributed since resources are numerous. However, different from the practical Web environment, existing method of the RBAC on the web is centralized. In this paper, we proposed a

practical Web security method using distributed role hierarchy. This system divides a global role hierarchy into several local role hierarchies. These local role hierarchies are managed by the local role hierarchy module. Proposed method removes the difficulties of role administration, thus the system will have proper and consistent role hierarchy. This method is also reliable against resource consumable network attack such as DoS. The approach can be easily applicable to any Web application, such as e-commerce and work-flow system for enhancing data and resource security.

## References

1. Ferraiolo, D., Kuhn, R.: Role-Based Access Control. Proceedings of 15th National Computer Security Conference. Balmy, Baltimore, USA (1992)
2. Ferraiolo, D., Kuhn, R.: An Introduction To Role-Based Access Control. NIST/ITL Bulletin. NIST (1995)
3. Ferraiolo, D., Barkley, J., Kuhn, D.: A Role Based Access Control Model and Reference Implementation within a Corporate Intranet. ACM Transactions on Information Systems Security, Vol. 1, No. 2. ACM (1999) 1-30
4. Barkley, J., Kuhn, D., Rosenthal, L., Skall, M., Cincotta, A.: Role-Based Access Control for the Web. CALS Expo International & 21st Century Commerce 1998: Global Business Solutions for the New Millennium (1998)
5. Freudenthal, E., Pesin, T., Port, L.: dRBAC: Distributed Role-based Access Control for Dynamic Coalition Environments. Proceedings of the 22nd International Conference on Distributed Computing Systems (2002)
6. Sandhu, R., Bhamidipati, V., Munawer, Q.: The ARBAC97 Model for Role-Based Administration of Roles. ACM Transactions on Information and System Security, Vol. 2, No.1 (1999) 105-135
7. Bauer, L., Schneider, M., Felten, E.: A General and Flexible Access-Control System for the Web. Proceedings of the 11th USENIX Security Symposium (2002)
8. Ungureanu, V., Vesuna, F., Minsky, N.: A Policy-Based Access Control Mechanism for the Coporate Web. 16th Annual Computer Security Applications Conference (2000)
9. Sirer, E., Wang, K.: An Access Control Language for Web Services. Proceedings of 7th ACM SACMAT (2002) 23-32
10. Sandhu, R., Coyne, E., Feinstein, H., Youman, C.: Role-Based Access Control Models. IEEE Computer, IEEE (1996) 38-47
11. Ferraiolo, D., Sandhu, R., Gavrilu, E., Kuhn, D., Chandramouli, R.: Proposed NIST Standard for Role-Based Access Control, ACM Transactions on Information and System Security, Vol4, No3 (2001) 224-274
12. Bacon, J., Moody, K., Yao, W.: Access Control and Trust in the use of Widely Distributed Services. Middleware 2001, LNCS 2218 (2001) 300-315
13. Tari, Z., Chan, S.: A Role-based Access Control for Intranet Security. IEEE Internet Computing (1997) 24-34
14. Park, J., Sandhu, R.: RBAC on the Web by Smart Certificates. Proceedings of the 4th ACM workshop on Role-based Access Control, ACM Press, (1999) 1-9
15. Ahn, G., Sandhu, R., Kang, M., Park, J.: Injecting RBAC to Secure a Web-based Workflow System. Proceedings of the Fifth ACM Workshop on Role-based Access Control (2000) 1-10

# A Low-Cost Checkpointing Scheme for Mobile Computing Systems

Guohui Li, Hongya Wang, and Jixiong Chen

College of Computer Science and Technology,  
Huazhong University of Science and Technology,  
Wuhan, Hubei, P. R. China, 430074  
{guohuili, andycjx}@263.net, hongyawang@sina.com

**Abstract.** In distributed computing systems, processes in different hosts take checkpoints to survive failures. For mobile computing systems, due to certain new characteristics conventional distributed checkpointing schemes need to be reconsidered. In this paper, we propose a low-cost coordinated checkpointing algorithm. During normal computation message transmission, the checkpoint dependency information among mobile hosts is recorded in the corresponding mobile support stations. When a checkpointing procedure begins, the initiator concurrently informs relevant mobile hosts, which minimizes the identifying time. Moreover, compared with existing coordinated checkpointing schemes, our algorithm blocks the minimum number of mobile support stations during the identifying procedure. Experimental simulation shows that the proposed algorithm outperforms other coordinated checkpointing schemes and can provide a better system performance for mobile computing systems.

## 1 Introduction

In distributed computing systems, processes in different hosts communicate with each other by messages. To survive failures, processes take checkpoints periodically or aperiodically. Although checkpointing techniques have been extensively studied for distributed computing systems in last decades, most of the previous work assumed that the systems are supported on relatively reliable networks<sup>[1][2]</sup>. As many other techniques in distributed computing systems, traditional checkpointing schemes can not be readily applied to mobile computing systems due to certain new characteristics of such systems, e.g., mobility, low bandwidth, disconnection, low power consumption and limited memory. Hence, efficient checkpointing algorithms for mobile computing systems need to be reconsidered<sup>[3][4]</sup>.

Basically, there are two kinds of checkpointing methods: the asynchronous and the coordinated. Overmany checkpoints and unexpected recovery time delay due to the limited bandwidth make the asynchronous checkpointing methods unsuitable for mobile computing systems<sup>[3]</sup>. In contrast, the coordinated methods have less storage and bandwidth need, need no garbage collection and have the characteristic of domino-free. The Koo-Toueg (KT) algorithm<sup>[5]</sup> only forces those processes that have

communicated with the checkpoint initiator either directly or indirectly since the last global checkpoint to take new checkpoints. While the identifying procedure of KT is sequential, which may result in long checkpointing latency in mobile computing systems. In [4], Cao and Singhal proposed an algorithm (CS) that makes use of mobile support stations to fulfill the identifying procedure. But there are several drawbacks in this algorithm. First, regardless of the number of mobile hosts involved in a global checkpoint, the identifying time is always approximately equal to three times as many as the average message latency between two mobile support stations and all mobile support stations must be blocked during identifying. Moreover, if there are plenty of mobile support stations, the proxy is prone to be bottlenecked due to too many request messages, which may result in longer identifying time latency. Finally, matrix multiplication is a time-consuming procedure if the number of mobile hosts is large and it may lengthen the identifying time as well.

In this paper a low-cost concurrent checkpointing algorithm is proposed. During normal message exchanges, the computation messages are transmitted along with checkpoint dependency information among mobile hosts and the location information of corresponding mobile support stations, which are used to update the data structures of each mobile host. When a checkpointing procedure begins, all relevant mobile hosts are informed concurrently using these data structures, which dramatically decreases the time latency of tracing dependency trees. Meanwhile, the proposed algorithm makes full use of the computation and storage ability of mobile support stations, which reduces synchronization message transmission overhead and power consumption of mobile hosts.

The rest of the paper is organized as follows. Section 2 presents preliminaries for coordinated checkpointing algorithms in mobile computing systems. Section 3 introduces a novel concurrent checkpointing algorithm. Section 4 presents the performance evaluation of the proposed algorithm. Finally, conclusion is made in section 5.

## 2 Preliminaries

A mobile computing system consists of a large number of mobile hosts (MHs) and relatively fewer static hosts called mobile support stations (MSSs). The MSSs are connected by a static wired network, which provides reliable FIFO delivery of messages. A cell is a geographical coverage area under an MSS. MHs can directly communicate with an MSS by a reliable FIFO wireless channel only if they stay in the cell supported by the MSS. A distributed computation consists of some processes running concurrently on fail-stop MHs or MSSs and message passing is the only way for processes to communicate with each other. Each process runs at its own speed and messages are exchanged through reliable communication channels, whose transmission delays are finite but arbitrary.

Due to inter-process communications, the state of a process may depend directly or indirectly on that of another process. If a process has to rollback due to failure, the other processes, which are directly or indirectly dependent on it, also have to rollback. In recovery, it is important that the system is recovered to a global consistent state

and continues its operations from this consistent state. A global state is consistent if it contains no orphan message, i.e., a message whose received event is recorded in the local state of the destination process, but its send event is lost in the local state of the source process<sup>[6]</sup>. In our checkpointing algorithm each checkpoint is associated with a unique sequence number. The sequence number of each process increases monotonically and the  $j^{\text{th}}$  checkpoint of process  $P_i$  is denoted as  $C_{i,j}$ . The sending and receiving events of message  $\tau$  are denoted as  $\text{send}(\tau)$  and  $\text{receive}(\tau)$  respectively.

**Definition 1.** If message  $\tau$  is sent from process  $P_i$  before it takes checkpoint  $C_{i,m}$ , we say that  $\text{send}(\tau) \in C_{i,m}$ . Similarly, if  $P_i$  takes checkpoint  $C_{i,m}$  before it sends message  $\tau$ , we say that  $\text{send}(\tau) \notin C_{i,m}$ .

**Definition 2.** If message  $\tau$  is received and processed by process  $P_j$  before  $P_j$  takes checkpoint  $C_{j,n}$ , we say that  $\text{receive}(\tau) \in C_{j,n}$ . and if  $P_j$  takes checkpoint  $C_{j,n}$  before it receives  $\tau$ , we say that  $\text{receive}(\tau) \notin C_{j,n}$ .

**Definition 3.** Direct Checkpoint Dependency - Suppose there are two processes  $P_i$  and  $P_j$  and the latest checkpoints of the two processes are  $C_{i,m}$  and  $C_{j,n}$  respectively.  $P_j$  is directly checkpoint dependent upon  $P_i$ , denoted as  $P_j \text{ DCD } P_i$ , if and only if:

$$\exists \tau, \text{receive}(\tau) \notin C_{i,m} \wedge \text{send}(\tau) \notin C_{j,n}$$

**Definition 4.** Transitive Checkpoint Dependency - Process  $P_j$  is transitively checkpoint dependent upon process  $P_i$ , denoted as  $P_j \text{ TCD } P_i$ , if and only if:

$$\exists P_k, (P_j \text{ DCD } P_k) \wedge ((P_k \text{ DCD } P_i) \vee (P_k \text{ TCD } P_i))$$

In the following discussion, DCD and TCD are both called checkpoint dependency (CD) unless they are referred to explicitly.

### 3 A Concurrent Checkpointing Algorithm

#### 3.1 Underlying Notions of the Algorithm

Suppose there are  $N_{\text{mh}}$  MHs and  $N_{\text{mss}}$  MSSs in the system, where  $N_{\text{mh}}$  is much larger than  $N_{\text{mss}}$ . For clarity of presentation, we assume there is only one process running on each MH. Then in the following sections  $\text{MH}_i$  and  $P_i$  are used to denote the same meaning, that is, process  $P_i$  running on  $\text{MH}_i$ .

In KT, the procedure of tracing the dependency tree is sequential since each process only knows processes that are DCD upon it, which may result in long checkpointing latency due to the limited wireless bandwidth and the search cost. The search cost is the cost to locate which MSS an MH stays in currently. In our checkpointing scheme, the checkpoint dependency information owned by the source process and its location information are sent to the destination process along with the computation messages during normal computation message exchanges. Using such additional information each process knows processes that are not only DCD but also TCD upon it, which may reduce the depth of dependency trees and the checkpointing latency.



Moreover, by using the location information an MH may be identified without incurring the search cost, which may further shorten the checkpointing latency. Regarding the limited power and computation ability of MHs, it is of great importance that making full use of the computation ability and power of MSSs. Therefore, for an MH, in our checkpointing scheme the checkpoint dependency information and other MHs' location information are maintained in its corresponding MSS instead of in itself. To maintain such additional information for MHs, each MH, say  $MH_i$ , is associated with three sets  $PS\_RMF_i$ ,  $PS\_SMT_i$ ,  $TRACE_i$  and a boolean variable  $PROCESSED_i$  at its current MSS.

In  $PS\_RMF_i$ , for each mobile host  $MH_k$  if  $MH_k \text{ CD } MH_i$  holds, there exists a 2-tuple  $\langle k, p \rangle$  in which:

$k$  is the identity number of mobile host  $MH_k$  that  $MH_k \text{ CD } MH_i$  holds;

$p$  is the identity number of the MSS in which  $MH_k$  has sent a message (directly or indirectly) to  $MH_i$ .

Similarly, in  $PS\_SMT_i$ , for each mobile host  $MH_k$  which has received at least one message from  $MH_i$ , there exists a 2-tuple  $\langle k, p \rangle$  in which:

$k$  is the identity number of mobile host  $MH_k$  on which  $MH_i \text{ CD } MH_k$  holds;

$p$  is the identity number of the MSS in which  $MH_i$  has sent a message to  $MH_k$  directly.

$TRACE_i$  records identity numbers of MSSs that  $MH_i$  has passed by.  $PROCESSED_i$  denotes whether  $MH_i$  has received a checkpointing request during a global checkpointing procedure. The three sets are all initialized as empty and  $PROCESSED_i$  is initialized as FALSE.

To maintain the correctness and consistency of sets  $PS\_RMFs$  and  $PS\_SMTs$ , message exchanges between MHs include the following extra steps:

(1) Suppose  $MH_i$  is in the cell of  $MSS_p$  and  $MH_j$  is in the cell of  $MSS_q$ . When  $MH_i$  sends out a message  $\tau$  to  $MH_j$ , it first sends the message to  $MSS_p$  and then  $MSS_p$  locates  $P_j$ 's corresponding mobile support station  $MSS_q$ . After inserting 2-tuple  $\langle j, q \rangle$  into  $PS\_SMT_i$ ,  $MSS_p$  sends the message to  $MSS_q$  together with  $PS\_RMF_i$  and 2-tuple  $\langle i, p \rangle$ .

(2) When  $MSS_q$  receives the message, it first extracts the attached information, and then forwards it to  $MH_j$ . Finally  $MSS_q$  acts as follows:

```

FOR each 2-tuple  $\langle k, m \rangle$  in  $PS\_RMF_i \cup \{ \langle i, p \rangle \}$ 
  IF  $k \neq w$  (  $w$  is the first item of any 2-tuple in
     $PS\_RMF_j$  )
     $PS\_RMF_j = PS\_RMF_j \cup \{ \langle k, m \rangle \}$ ;
  ELSE
    IF  $m \neq n$  (  $n$  is the second item of any 2-tuple in
       $PS\_RMF_j$  whose  $w = k$  )
       $PS\_RMF_j = (PS\_RMF_j - \{ \langle w, n \rangle \}) \cup \{ \langle k, m \rangle \}$ ;

```

### 3.2 Handling Mobility and Disconnection

Through above message exchanges, an MH, say  $MH_i$ , will partially or totally know which MHs are CD upon it according to the content of set  $PS\_RMF_i$ . At the same time, the location information of such MHs is also available. While in mobile computing systems, MHs may not always stay in only one MSS and if the handoff of one of such MHs occurs, its location information in  $PS\_RMF_i$  may become out of date. To cope with handoffs the following strategy is adopted. When the handoff of an MH, say  $MH_i$ , occurs, the handoff history is recorded into  $TRACE_i$ . If the number of handoffs becomes bigger than a predefined threshold, position-changed messages are broadcasted. The main purpose of the threshold is to avoid too many position-changed messages being generated if  $MH_i$  frequently moves around in several relatively fixed MSSs that are adjacent to each other. The detailed handoff processing procedure is described as follows.

When  $MH_i$  leaves a cell and enters into another cell, it first ends its current connection by sending a  $LEAVE(MHID)$  message to the old MSS, say  $MSS_p$ , and then establish a new connection by sending a  $CONNECT(MHID, OLDMSSID)$  message to the new MSS, say  $MSS_q$ . After receiving the  $CONNECT$  message,  $MSS_q$  sends a  $GETDATA(MHID)$  message to  $MSS_p$  to fetch  $MH_i$ 's data structures. Once  $MSS_q$  obtains all data structures for  $MH_i$ , it inserts  $MSS_p$ 's identity number, namely  $p$ , into  $TRACE_i$ .  $MSS_p$  will release all resources occupied by  $MH_i$  after finishing relevant data structures transmission.

If  $MSS_q$  finds that the number of items in  $TRACE_i$  has exceeds a predefined threshold, it will send position-changed messages to all MSSs in  $\Pi_2(PS\_SMT_i)$ , along with  $\Pi_2(PS\_SMT_i)$  and 2-tuple  $\langle i, q \rangle$  to minimize the amount of position-changed messages.  $\Pi_i(PS\_SMT)$  is such a set that consists of the  $i^{th}$  item of each 2-tuple in  $PS\_SMT$ .  $TRACE_i$  is set as empty after  $MSS_q$  finishes the position-changed messages transmission. When an MSS, say  $MSS_r$ , receives a position-changed message, it acts as follows:

IF having received the same position-changed message  
Return;

FOR each  $PS\_RMF_j$  in  $MSS_r$

FOR each 2-tuple  $\langle k, m \rangle$  in  $PS\_RMF_j$

IF  $k = i$

$m = q$ ;

FOR each  $PS\_SMT_j$  in  $MSS_r$

FOR each 2-tuple  $\langle w, n \rangle$  in  $PS\_SMT_j$

IF  $n \notin MSSs\_SET_{rec}$

Send the position-changed message to  $MSS_n$  along

with  $MSSs\_SET_{rec} \cup ( \bigcup_{MH_i \in MSS} \Pi_2(PS\_SMT_i) )$  and 2-tuple  $\langle i, q \rangle$ ;

$MSSs\_SET_{rec}$  is the set of MSSs' identifiers along with the received position-changed message and can be used to minimize the number of position-changed messages.

Disconnection is another problem that may affect the performance of a checkpointing algorithm for mobile computing systems. In our checkpointing scheme, techniques used to cope with disconnection in [4] are adopted. When an MH, say  $MH_i$ , wants to disconnect from its local MSS, say  $MSS_p$ , it takes a local checkpoint and transfers its local checkpoint into  $MSS_p$ . If  $MH_i$  is asked to take a checkpoint during disconnect interval,  $MSS_p$  is responsible for convert the local checkpoint into new permanent checkpoint of  $MH_i$ .

### 3.3 Algorithm

In a mobile computing system, to minimize the number of MHs taking checkpoints, instead of forcing all MHs to take checkpoints, the global checkpoint initiator has to identify which MHs must take a checkpoint and the identifying procedure is not trivial. The detailed checkpointing algorithm is described as follows:

(1) When mobile host  $MH_i$  initiates a checkpointing procedure, it first sends a request message to its local MSS, say  $MSS_p$ , and then  $MSS_p$  becomes the checkpointing proxy that is responsible for the entire procedure. After receiving the request  $MSS_p$  acts as follows:

```

TEMP =  $\Pi_2(PS\_RMF_i)$ ;
weight = weight / |TEMP|;
/* weight is the value to be assigned to should- take-
checkpoint MHs. It is mainly used to determine the ter-
mination of the checkpointing procedure [7] and is initi-
ated as 1*/
FOR each q belongs to TEMP
    Send checkpointing request <  $MH_{ini}$ ,  $MSS_{ini}$ , weight,
 $MH_{List}$  > to  $MSS_q$ , along with  $PS\_RMF_i$  to minimize the num-
ber of request messages among MSSs;

```

Where  $MH_{ini}$  is the identity number of the initiator, namely  $i$ ;  $MSS_{ini}$  is the identity number of the checkpointing proxy, namely  $p$ ; TEMP is initiated as empty;  $MH_{List}$  is the identity number list of should- take-checkpoint MHs in the cell of  $MSS_q$ ;  $\delta_{2=q}(PS\_RMF)$  is a subset of  $PS\_RMF$  in which the second item of each 2-tuple is equal to  $q$ .

(2) When  $MSS_q$  receives checkpointing request < $i$ ,  $p$ , weight,  $MH_{List}$ >, it acts as follows:

```

FOR each  $MH_j$  whose identity number belongs to  $MH_{List}$ 
    IF  $MH_j$  is in the current cell
        IF PROCESSEDj  $\neq$  TRUE

```

```

PROCESSEDj = TRUE;
Blocking the application message transmission of MHj;
FOR each 2-tuple <w, r> in PS_RMFj
  IF r ∉ Π2(PS_RMFrec)
    TEMP = TEMP ∪ {r};
  ELSE
    IF (Π1(δ2=r(PS_RMFj) - Π1(δ2=r(PS_RMFrec))) ≠ ∅
      TEMP = TEMP ∪ {r};
weight = weight / (|TEMP| + 1);
FOR each MSSr whose identity number belongs to TEMP
  Send checkpointing request <i, p, weight,
  Π1(δ2=r(PS_RMFj) - Π1(δ2=r(PS_RMFrec)))> to MSSr, along with
  PS_RMFrec ∪ ( ∪j ∈ MHList PS_RMFj );
weight = weight / |MHList|;
FOR each MHj whose identity number belongs to MHList
  IF MHj is in current cell
    Send checkpointing request <i, p, weight, ∅> to MHj;
  ELSE
    Locate MHj's corresponding MSS, say MSSs, and send
    MSSs checkpointing request <i, p, weight, j> along with
    PS_RMFrec ∪ ( ∪j ∈ MHList PS_RMFj );

```

Where PS\_RMF<sub>rec</sub> is set PS\_RMF associated with the received request message, which is used to minimize the number of request messages among MSSs. TEMP is initiated as empty.

(3) After receiving a request message, each MH sends a positive or negative reply message to MSS<sub>p</sub> along with the weight in the request message. For an MH, it may receive more than one checkpointing requests with different weights. To ensure the initiator can determine the termination of a global checkpointing procedure, it must send reply messages along with the sum of different weights to the initiator.

(4) When MSS<sub>p</sub> receives the votes from all relevant MHs, namely the sum of weights in reply messages is equal to 1, it makes a positive or negative decision and broadcasts the decision to all involved MHs, and then all involved-in MHs act accordingly. This is almost the same as any other two-phase strategies.

(5) If all MHs involved in the global checkpoint have turned their tentative checkpoints into permanent ones, their data structures, PS\_RMFs, PS\_SMTs and TRACES, are all set as empty. Note that if TRACE is not empty, the position-changed messages sending procedure is called explicitly. PROCESSEDs are set as FALSE as well.

## 4 Experimental Evaluation

We design and implement a set of experiments evaluate these three algorithms by performance metrics *the average identifying time, the total application message*

*overhead* and *the average MSS blocking time*. The average identifying time is defined as the sum of the identifying time of all global checkpoints divided by the number of global checkpoints. The total application message overhead denotes the message transmission cost for transmitting computation messages among MHs during their normal execution plus the position-changed message transmission cost among MSSs. The average MSS blocking time is defined as the total blocking time of all MSSs due to checkpointing divided by the number of MSSs and the number of all global checkpoints during a certain period of time.

In our simulation system, there are 10 MSSs, each of which has a wireless communication with the mobile hosts currently in its cells. We vary the number of MHs from 10 to 100 to see the corresponding changes of these performance metrics. The wireless LAN has a bandwidth of 100Kbps which follows IEEE 802.11 standard<sup>[8]</sup>. The bandwidth for the wired network is 10 Mbps. The incremental checkpoint<sup>[1]</sup> is used to reduce the amount of data that must be written to the stable storage. Disconnections are dealt with according to the techniques in [4]. The time span between two consecutive global checkpoints is 100s and a random active process is selected as the global checkpoint initiator at the scheduling time point. Each process sends out a computation message to another randomly selected active process after a time period that follows an exponential distribution and  $\lambda_{\text{sendmessage}}$  is equal 0.1. Disconnection rate and handoff rate both follow the Poisson process and  $\lambda_{\text{disconnection}}$  and  $\lambda_{\text{handoff}}$  are both equal to 0.01. The size of each computation message is 1K, which does not include the additional information in CCA. The threshold of sending position-changed messages is equal to 3.

In experiments, we change the number of mobile hosts and for each case the system runs for half an hour. Figure 1 illustrates the total application message overheads as a function of the number of MHs. The basic unit of Y-axis is the total application message overhead for KT where there are 10 MHs. From Figure 1 we can see that the total application message overheads for KT, CS and CCA are almost identical and the overhead of CCA is a little higher than that of the others. The reason lies in that additional information is transmitted during the normal computation message exchanges in CCA and position-changed messages may be generated if the number of handoffs of an MH exceeds the predefined threshold.

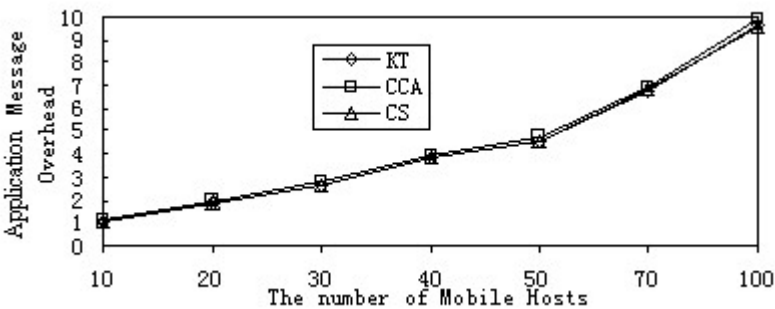


Fig. 1. Total Application Message Overhead Comparison

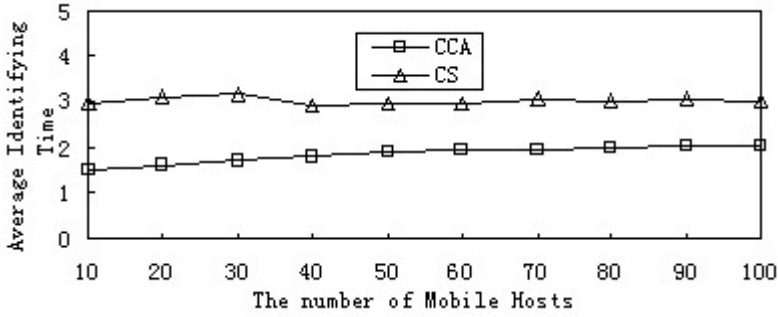


Fig. 2. Average Identifying Time Comparison

The average identifying time is an important performance metrics, especially for certain time-critical service-oriented applications. Figure 2 illustrates the average identifying time for CS and CCA and the basic unit of Y-axis is the average MSS-to-MSS message latency. KT is not depicted because of its obvious shortages. As depicted in Figure 2, with the number of MHs increases, for CS, the average identifying time is approximately three times as many as the average MSS-to-MSS message latency and varies slightly with the variation of the number of MHs. For CCA, the performance metric outperforms that of KT and CS because the involved-in MHs are informed concurrently and the search cost decreases thanks to the additional information maintained in MSSs. Though the average identifying time of CCA has a slight ascending trend with the number of MHs increases, we can see that as long as the message communication is relatively intensive CCA will do better than CS.

Figure 3 illustrates the average MSS blocking time for CS and CCA. KT is not depicted because of its obvious shortages. The basic unit of Y-axis is the average MSS-to-MSS message latency. As shown in Figure 3, the average MSS blocking time of CCA is significantly smaller than that of CS, which is approximately equal to twice as many as the average MSS-to-MSS message latency. At the same time, the average MSS blocking time of CCA decreases accordingly with the number of MHs increases. The main reason is that in CCA only the minimum number of MSSs are blocked during every global checkpointing procedure while in CS all MSSs are blocked. The descending trend of the average MSS blocking time for CCA can be argued that with the number of MHs increases, for MHs in different MSSs, the probability of being

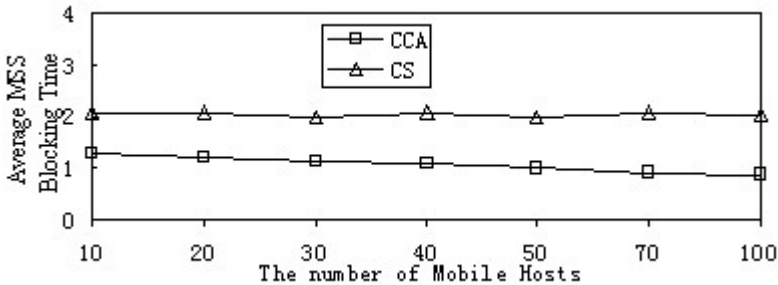


Fig. 3. Average MSS Blocking Time Comparison

checkpoint dependent upon each other becomes less due to the constant  $\lambda_{\text{sendmessage}}$  and then the number of MSSs involved in a global checkpointing procedure decreases probabilistically.

## 5 Conclusion

In order to survive a failure, processes in distributed systems have to take checkpoints to reduce the loss of computation. In mobile computing systems, due to the intrinsic characteristics of the wireless network and the mobile hosts, the checkpointing algorithm should be reconsidered deliberately. This paper proposes a novel concurrent checkpointing algorithm. By recording the dependency relation among mobile hosts and the location information of mobile support stations during the normal computation message transmission, the algorithm can reduce the time latency for a global checkpointing procedure significantly. Furthermore, it minimizes the number of blocked mobile support stations during identifying, which improves the system performance compared with previous coordinated checkpointing algorithms. The experimental simulation reveals that the proposed algorithm has better performance than existing ones such as KT and CS.

## Acknowledgements

The work in this paper was partially supported by the National Nature Science Foundation under grant 60203017 and SRF for ROCS, SEM.

## References

1. Deng, Y., Park, E. K.: Checkpointing and rollback-recovery algorithms in distributed systems. *Journal of Systems and Software*, Apr. (1994) 59-712
2. Elnozahy, E. N., Johnson, D. B., Zwaenepoel, W.: The performance of consistent checkpointing. *Proc. 11th Symp. Reliable Distributed Systems*, Oct. (1992) 86-95
3. Prakash, R., Singhal, M.: Low-cost checkpointing and failure recovery in mobile computing systems. *IEEE Tran. Parallel and Distributed Systems*, Oct., (1996) 1035-1048
4. Cao, G.H., Singhal, M.: On the Impossibility of Min-Process Non-Blocking Checkpointing and An Efficient Checkpointing Algorithm for Mobile Computing Systems. *Proc. The 27th Intl. Conf. On Parallel Processing*, Aug. (1998) 37-44
5. Koo, R., Toueg, S.: Checkpointing and roll-back recovery for distributed systems. *IEEE Tran. On Software Engineering*, Jan. (1987) 23-31
6. Randell, B.: System structure for software tolerance. *IEEE Transactions on Software Engineering*, Jun. (1975) 220-232
7. Huang, S. T.: Detecting termination of distributed computations by external agents. *Proc. 9th International Conf. Distributed Computing Systems*, Jun., (1989) 79-84
8. Crow, B., Widjaja, I., Kim, J., Sakai, P.: IEEE 802.11 Wireless Local Area Networks. *IEEE Comm. Magazine*, Sept. (1997) 116-126

# Semantic and Adaptive Middleware for Data Management in Smart Vehicle Space\*

Qing Wu, Zhaohui Wu, Bin Wu, and Zhou Jiang

College of Computer Science, Zhejiang University,  
Hangzhou, Zhejiang, China 310027  
{wwwsin,wzh,branwu,z8j1}@zju.edu.cn

**Abstract.** With the increasing demands for adaptive middleware of real-time embedded systems in ubiquitous computing environments, the need for novel software architecture and programming infrastructure to manage data is widely recognized. In this paper, we present SAM (Semantic and Adaptive Middleware), an efficient middleware architecture that supports for real-time embedded system in smart vehicle space. The SAM's architecture is based on embedded real-time OS and CAN-based communication environment, which includes tinyORB kernel, distributed components, distributed meta objects, services and tools. In SAM, we mainly utilize context-sensitive repository and knowledge management technology to realize adaptive characteristic. The two contributions of this paper are the proposal of SVA (Semantic Virtual Agent) and component-level adaptive mechanism. The SVAs use SIP (Semantic Interface Protocol) to communicate and auto-update themselves. To achieve self-adaptation and reflection of component-level, the component hook mechanism is introduced. In addition, we have brought into further discussion of the hardware infrastructure and one application of SAM to manage heterogeneous data.

## 1 Introduction

In 21st Century, ubiquitous computing would let us “off the desktop,” and make us go into the new world where all entities in environments fuse naturally [1]. People live in the intelligent active environments, where they using smart devices to interact and seamlessly integrate with each other naturally. The active environments, smart devices, and people are essential elements of ubiquitous computing environments, and they continuously adjust themselves to better cooperation [2]. As a result, it poses a number of new challenges for middleware technology. In particular, such user-centered ubiquitous computing environments may require a semantic and adaptive middleware architecture to trustily communicate, operate resources, manage data, and provide various services for real-time vehicle space [3][4].

---

\* This research was supported by 863 National High Technology Program under Grant No. 2003AA1Z2080, “Research of Operating System to Support Ubiquitous Computing”



The traditional middleware is inevitably heavyweight, has not enough reconfigurable mechanisms and manipulating policies [5], and also has not specifically semantic interface, which is not suitable for ubiquitous computing such as in vehicle space. In addition, the increasing applications in the vehicle space would require the middleware to execute more efficiently and reliably. At present, many research efforts have focused on designing new middleware architecture capable of supporting the requirements imposed by ubiquitous computing [6][7][8]. PICO [9] supports time-critical applications in dynamic, heterogeneous environments. Gaia [10] coordinates software entities and heterogeneous networked devices contained in active physical space, exporting services to query and utilize existing resources. Open ORB [11] middleware provides component frameworks and reflection mechanism used to discover the current structure and behavior. Recently, the vehicles have merged into our daily life, so we select the vehicle space as a representative scene of ubiquitous computing. However, in ubiquitous computing environments such as smart vehicle space, in our opinion, a more systematic, hierarchy, semantic-integrated and adaptive solution is needed. Therefore, the motivation behind our research is the lack of a suitable middleware software infrastructure to develop applications for ubiquitous computing. In this paper, we investigate the semantic and adaptive middleware architecture for smart vehicle space in details.

The structure of the paper is as follows. Section 2 presents SAM infrastructure, highlighting smart vehicle space. Next, the hierarchy, semantic and adaptive middleware architecture is given. Section 3 then introduces SVA in detail. In particular, the key technology underpinning of SVA, namely SIP is proposed. Section 4 argues component-level adaptive mechanism in SAM. Specially, the section 3 and section 4 put forward the application of knowledge management in SAM. Following this, section 5 discusses the application of SAM's hardware infrastructure and gives a case study. Finally, section 6 summarizes the discussion and introduces some areas demanding further investigation.

## 2 SAM Infrastructure

SAM consists of smart vehicle space and hierarchy architecture. After people go into smart vehicle space, they can communicate with the embedded equipments naturally through SAM infrastructure. Hereinafter, we give a detailed introduction of above two aspects.

### 2.1 Smart Vehicle Space

In recent years, lots of developers have applied embedded technology to vehicles. The drive capability, dependability, comfort, and convenience of the vehicle are improved greatly [12]. When people enter into smart vehicle space, they could find many intelligent devices and equipments around them. People want to communicate with these devices naturally and friendly. They also want to form a harmonious entia. The smart vehicle space, surrounding people and devices, provides easy communica-

tion and processing transaction environments where we have used synthetical technology such as computer communication technology, advanced human-machine interface, auto control, positioning and knowledge management technology, etc.

Smart Vehicle Space is composed of (1) auto apperceiving context system, (2) auto processing system, (3) context repository reasoning system and (4) centralized controlling system, as shown in Figure 1.

Auto apperceiving context system consists of the status of people and equipments in the vehicle, including cameras, sensors, and sound receivers. Auto processing system comprises steering, navigating and entertainment subsystem. Context repository reasoning system uses correlative context and knowledge management technology to make manipulating strategy. Particularly, the centralized controlling system is kernel of the smart vehicle space, controlling other parts to communicate and cooperate effectively.

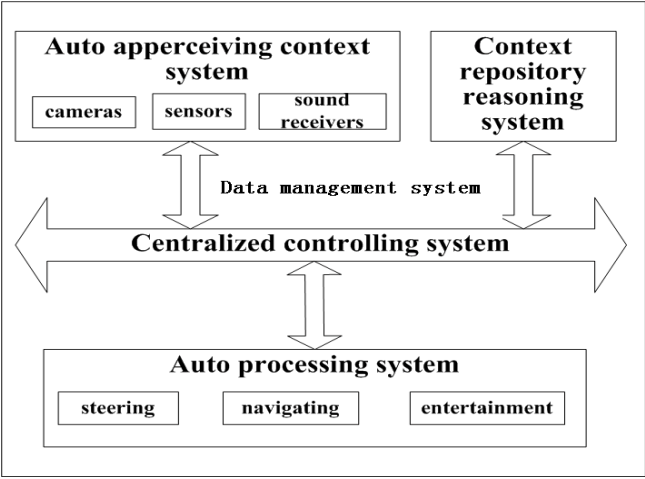


Fig. 1. Smart Vehicle Space

2.2 SAM Architecture

The overall goal of SAM is to develop a semanteme-integrated and dynamic re-configurable middleware technology through a synthesis of reflection, component, and knowledge management technologies. In particular, SAM is structured as tiny-ORB kernel, a set of configurable distributed components, meta objects, and SVAs. The SAM processes transactions through interplay of SVA and context repository. SAM watches the structure and behavior of components, meta objects, SVAs, and context repository momentarily, by which SAM could change each part at run-time on principle. One of the key aspects of the SAM architecture is the communication of SVAs using SIP. In addition, we introduce the component manager that records structure and behavioral aspects of each component, managing component's life cycle. For example, before creating or destroying a component instance, it should be through component manager first which decides what and how to execute it.

SAM is made up of four layers, as Figure 2 shows. The physical layer consists of the hardware, associated drivers, and operating system kernel. The next layer comprises CAN-based communication environment such as J1939, CANOpen protocol and communication modules for above layer. The third layer is tinyORB kernel that is reduced from common ORB and supports dynamic characteristic. In this paper, we don't discuss tinyORB in detail. The fourth layer is the essential services and tools, including components, meta objects, and SVAs. SVA is the logic presence of group of services, which cooperate with each other by SIP. The meta object includes QoS(quality of service), naming, discovery, and event service. In addition, the tools of this layer consist of system monitoring and energy saving, etc.

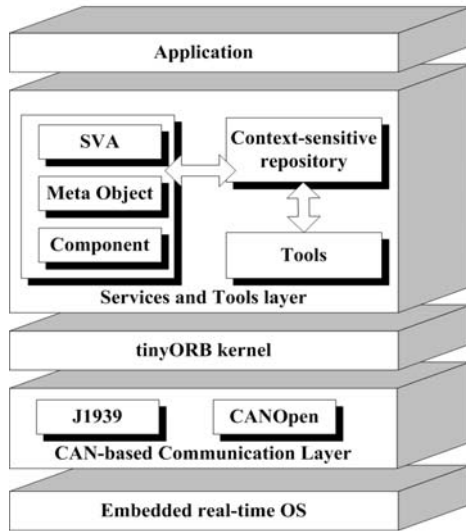


Fig. 2. SAM Architecture

### 3 SVA Framework

SVA is based on distributed components and distributed meta objects. The communication of SVAs use SIP, which consist of SVAs request, response and self-updating protocol. We utilize context repository reasoning system to update SVA's knowledge base and capability repository. At the same time, the SVA changes its missions on principle.

#### 3.1 SVA

In our design, we break the user's application into several tasks. Each task should be transacted by some SVAs. SVA is just a logic abstract union which comprises a number of related meta objects which are kinds of various services. The meta object has direct interface with components. Emphatically, the basic element is the compo-

nent. The relationships between two components may be mutex or interdependently. In SAM, component manager keeps each component's information and inter-relationship repository, which also organizes the meta objects using some strategy. The SVA infrastructure is shown in figure 3.

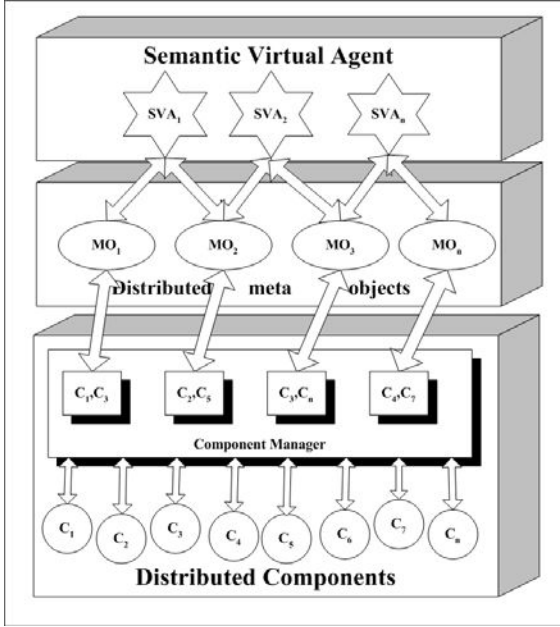


Fig. 3. SAM infrastructure

For example, one task needs  $SVA_1$ ,  $SVA_2$ , and  $SVA_3$ .  $SVA_1$  consists of  $MO_1$  and  $MO_2$ .  $SVA_2$  comprises  $MO_2$  and  $MO_3$ .  $SVA_3$  is made up of  $MO_3$  and  $MO_n$ .  $MO_1$  communicates with component manager, and passes the requirement semantic parameters. The component manager is in charge of looping up the related components and returning results. In this case,  $MO_1$  consists of  $C_1$  and  $C_3$ . Particularly, the SVA layer communicates with meta object layer and meta object layer communicates with component layer both use semanteme. In order to describe it, we present SIP as follows.

### 3.2 SIP

Semantic interface protocol is the rule of SVA's presence, development, and inter-communication, which includes SVA's naming, request, response, and self-updating protocol. In this section, we introduce SVA's naming protocol first. Here, we can define a SVA by  $P=\langle C, M, K \rangle$ , where  $C$  is the set of capabilities of this SVA;  $M$  is the set of SVA mission;  $K$  is the knowledge base of this SVA. For instance,  $SVA_1$  looks up and obtains one performance parameter capability and its missions are providing  $SVA_2$ 's position for  $SVA_3$ . Considering security and the right of each SVA, we

should know who this SVA could serve. In particular, the SVA use whose capability depending on the context repository reasoning system and its knowledge base. So, we can define  $SVA_i = \langle \{lookup, getonepara\}, \{ProvPos(SVA_2), SVA_3, \{k_1, k_2, \dots\}\} \rangle$ . As follow, we propose SVA request, response, updating knowledge base, and updating capability protocol.

### 3.3 SVA Request and Response

The communications of SVAs include SVAs request and response. One SVA can ask other SVA for help which is called SVA request. For example,  $SVA_1$  wants to get one parameter of the system. According as its knowledge base,  $SVA_1$  finds  $SVA_2$  has this capability. Then  $SVA_1$  sends request command to  $SVA_2$ , which is defined as Request  $(SVA_1, SVA_2) = \{(getonepara)\}$ . As for  $SVA_2$ , if it has not this capability now, it would find other SVA through its knowledge base. The process would not stop until one SVA has or all SVAs have not. In this case,  $SVA_2$  finds it has not, then it requests for  $SVA_3$ . But  $SVA_3$  also has not. Next according as  $SVA_3$ ' knowledge base, it finds  $SVA_7$ . Because  $SVA_7$  has this capability,  $SVA_7$ , not  $SVA_2$  would send result to  $SVA_1$ , and remind  $SVA_1$  to update its knowledge base. This is distributed peer to peer process. As shown in figure 4.

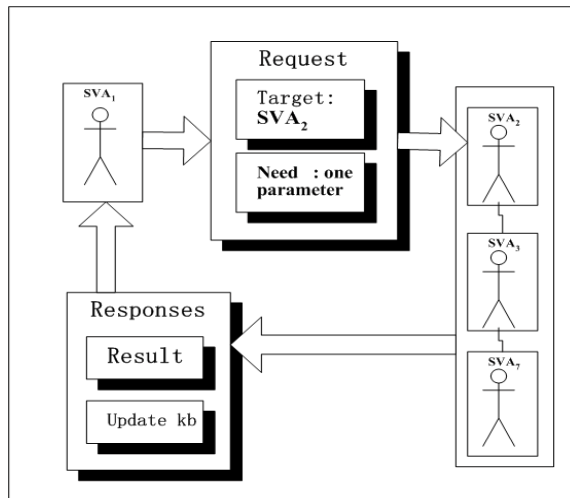


Fig. 4. SVA request and responses

### 3.4 SVA Self-updating

SVA is developing continuously in its life cycle. In one SVA's life cycle, it can acquire new knowledge from other SVAs and context from smart vehicle space, using context repository reasoning system to update its knowledge base. SVA would also obtain new capabilities and update its capability repository. Of course, the SVA

would change its missions constantly. SVA is adjusting and developing ceaselessly which is an adaptive process. In addition, SVA provides its knowledge actively and accept other's knowledge to update its knowledge base. Here, for instance, we can define  $\text{UpdateKb}(\text{SVA}_i)$  by  $\text{Acquire Knowledge}(\text{SVA}_i) + \text{UpdateKb}$ . The SVA could also discern its status by its knowledge base. Once finding it has a new capability, the SVA would update its capability repository. We can define  $\text{UpdateCapability}(\text{SVA}_i)$  by  $\text{DiscernStatus}(\text{SVA}_i) + \text{UpdateCap}$ .

## 4 Component-Level Adaptive Mechanism

The base of the SAM infrastructure is the distributed components layer. Components are the tiny self-managing, autonomic parts of the system which work in distributed environments [13]. As Szyperski said, a component can be defined as “a unit of composition with contractually specified interfaces and explicit dependencies only”. In particular, he argues “a software component can be deployed independently and is subject to composition by third parties” [14]. In SAM, we emphasize the organization of the components to realize the reuse ability. The component's life cycle and the relationship of components are administered by component manager [15]. As a result, we can realize the component reconfiguration in runtime depending on component manager.

### 4.1 Reflection

Reflective technology has been applied to language design, such as Java Core Reflection API. Recently, reflection has also adopted in distributed operating system and middleware design. In particular, reflection allows components to customize their structure and behavior [16]. The styles of reflection consist of (1) structural reflection and (2) behavioral reflection. Structural reflection includes the set of interfaces supported. Behavioral reflection is concerned with actions and intentions in the underlying system. The main motivation for our research is to integrate the component technology into reflective technology. In our design, we use component hook mechanism to realize it.

### 4.2 Component Hook Mechanism

Component manager is the central unit in component group. When the meta object wants to create or destroy one instance of the basic component, it must inform component manager through component hook first. Component manager records the status of each basic component instance, and has capability to terminate one basic component instance on condition. For example, if component manager decides component 1, component 2, and component 3 make up of meta object 1. First component manager create these component instances and discerns their status at any moment. When the context of meta object 1 change, component manager will reconfigure the

components combination. First, component manager should judge which component is not needed and has not correlation with remainder components, then removes this component and relative components. Next, component manager should judge which component is required to be appended, and also add the component which depends on. For instance, component 2 and component 3 are not needed, but component 1 relies on component 3, then only component 2 is removed. Additionally, component manager judges component 4 is needed, and which depends on component 6, then component 4 and component 6 are both added. Then the component creating and destroying processes are executed by component manager.

## 5 Application of the SAM

SAM is a combination of communication, embedded, intelligent agent, and knowledge management technology. To achieve the goal of SAM, we have integrated above four key technologies to implement the SAM architecture which includes hardware layer, the software layer, and the smart vehicle space.

### 5.1 Hardware Framework for SAM

In SAM, the hardware infrastructure consists of (1) Central Control Unit (CCU), (2) Electronic Control Unit (ECU), (3) Smart Sensor, and (4) Smart Actuator. CCU is the administrator of the hardware architecture which controls and manages other parts. ECU can gain instructions for actuators by analyzing the data collected by sensors and transfer the data offered by sensors to the network or get instructions from the network. ECU can gain instructions for actuators by analyzing the data collected by sensors and transfer the data offered by sensors to the network or get instructions from the network. Smart Sensor is the information collection equipment depending on industry buses comprising kinds of convertors and sensors. Different with traditional sensor, the Smart Sensor has the capability of digit communication which can self-manage without the interruption of center processor. On the contrary, Smart Actuator is executable equipment cooperating with other Actuators. They use CBL (CAN-based network layer) to communicate with each other. The CAN (control Area Net) is a serial communications protocol which efficiently supports distributed real-time control with a very high level of security. CBL provides objects, protocols, and services for the event driven or requested transmission of CAN messages and for the transmission of larger data blocks between CAN devices. Furthermore, CBL offers mechanisms for the automatic distribution of CAN identifiers and for the initialization and monitoring of nodes. As a result, they constitute smart vehicle ad hoc network.

### 5.2 Our Going Work

To illustrate the application of SAM, we give our going work as follow. We assume when the host is driving the car, the SAM would start watch-status SVA to recognize

host's face, sound, and actions. If this SVA feel host tried, it would alarm host to drive slow. Even this SVA can stop the car compulsorily before the emergency might occur. When this SVA feel the host oppressive, it would start entertainment SVA to play music. If the host sits uncomfortably, the watch-status SVA would start adjust-seat-height SVA to fit adaptive position. In particular, the navigating SVA cooperates with planning SVA to remind host when, what, where and how to do at times. For example, the navigating SVA could tell host it is time to go to office for a meeting. When the host arrived at office, he is indicated by stop-car SVA to park. Then the stop-car SVA could start the following task: open the car door, close the car door, turn off the Radio, and close the air conditioner. To finish these tasks above, all equipments including electrical car door, electrical seat, air conditioner, CD player, GPS and GIS should cooperate with each other intelligently and all their data should be managed effectively, which is a great task.

## 6 Conclusion and Future Work

We have proposed the Semantic and Adaptive Middleware Architecture for Smart Vehicle Space that supports knowledge management in pervasive environments. The design of the SAM framework is based on embedded real-time OS, tinyORB and CAN-based communication layer. This paper focuses on synchronization and adaptability aspects of Smart Vehicle Space. We bring forward the Smart Vehicle Space and SAM architecture. Further we argue that it is crucial to focus on the human-centered aspects of interaction and communication. The faith of the service should include providing actively, depending on context, continuously adjusting and component integrated. The Semantic Interface Protocol is presented in detail which focused on interaction between the Semantic Virtual Agents. Next, we give an application of SAM, from a practical point of view; the prototype implementation is expected to yield a higher level of quality of embedded smart devices and the integration degree of Smart Vehicle Space.

This is however a preliminary analysis and further work remains to be done to test SAM architecture more fully. Our future work includes (1) Extending the Semantic Interface Protocol for communication between SVAs (2) Studying more advanced mechanisms for Semantic and Adaptive middleware (3) Finding other active space except Smart Vehicle Space (4) Integrating biology authentication and machine learning technology to provide more natural, context-sensitive adaptation.

## References

1. Weiser M. The Computer for the 21st Century. *Scientific American*. (1991) 94-100
2. M. Satyanarayanan. A Catalyst for Mobile and Ubiquitous Computing, Vol 1. *Pervasive Computing*. (2002) 2-5
3. Anand Tripathi. Next-Generation Middleware Systems Challenges Designing. *Communications of The ACM* Vol. 45, No. 6. (2002) 39-42



4. Kimmo Raatikainen, Henrik Bærbak Christensen, Tatsuo Nakajima. Application Requirements for Middleware for Mobile and Pervasive Systems. *Mobile Computing and Communications Review*, Volume 6, Number 4. (2002) 17-24
5. Kay Romer, Oliver Kasten, Friedemann Mattern. Middleware Challenges for Wireless Sensor. *Mobile Computing and Communications Review*, Volume 6, Number 4 Networks. (2002) 59-61
6. Thomas Phan, Richard Guy, Rajive Bagrodia. Scalable, Distributed Middleware Service Architecture to Support Mobile Internet Applications. *Workshop on Wireless Mobile Internet 7/01 Rome, Italy*. (2001) 27-33
7. E. Saridogan. Design and Implementation of a Concurrent, Object-Oriented, Real-Time and Distributed Programming Language with its Supportive Run-Time System. Ph.D. Thesis. (2000)
8. L. Bergmans, A. van Halteren, L. Ferreira Pires, M. van Sinderen, M. Aksit. A QoS-Control Architecture for Object Middleware. Application of Middleware in Services for Telematics project. (2000) <http://amidst.ctit.utwente.nl>
9. Mohan Kumar. Proc. Pervasive Information Community Organization (PICO) of Camileuns and Delegates for Future Global Networking. *LSN Workshop*. (2001) <http://www.ngi-supernet.org/lsn2000/UT-Arlington.pdf>
10. Manuel Román, Christopher Hess, Renato Cerqueira, Anand Ranganathan, Roy H. Campbell, Klara Nahrstedt. Gaia: A Middleware Platform for Active Spaces. *Mobile Computing and Communications Review*, Volume 6, Number 4. (2002) 65-67
11. Gordon S. Blair, Geoff Coulson, Lynne Blair. Reflection, Self-Awareness and Self-Healing in OpenORB. *Workshop on Self-Healing Systems '02*, Nov 18-19, 2002, Charleston, SC, USA. (2002) 9-14
12. Guoqing Yang, Zhaohui Wu, Xiumei Li, Wei Chen. SVE: Embedded Agent Based Smart Vehicle Environment. *Intelligent Transport System Conference 2003*. (2003) 1745-1749
13. Aniruddha Gokhale, Douglas C. Schmidt, Balachandran Natarajan, Nanbor Wang. Applying Model-Integrated Computing to Component Middleware and Enterprise Applications. *Communications of The ACM* October 2002/Vol. 45, No. 10. (2002) 65-70
14. Clemens Szyperski. *Component Software - Beyond Object-Oriented Programming* Addison-Wesley/ACM Press. (1998)
15. Scott M. Lewandowski. Frameworks for Component-Based Client/Server Computing. *ACM Computing Surveys*, Vol. 30, No. 1, March 1998. (1998) 3-27
16. Fabio Kon, Fabio Costa, Gordon Blair, Roy H. Campbell. The Case for Reflective Middleware. *Communications of The ACM* June 2002/Vol. 45, No. 6. (2002) 33-38

# Dynamic Resource Reservation Using the Differentiated Handoff Estimation Model for Mobile Networks

Si-Yong Park and Ki-Dong Chung

Dept. of Computer Science, Pusan National University,  
Kumjeong-Ku, Busan 609-735, Korea  
{syPark,kdchung}@melon.cs.pusan.ac.kr

**Abstract.** In this paper, we propose a differentiated handoff estimation model and a dynamic resource reservation scheme. The differentiated handoff estimation model is modeled with the reducible Markov chain based on inner regions in a cell.

The dynamic resource reservation scheme can dynamically control already reserved resource by change of the mobility class of a mobile host. And the scheme can reserve resource for mobile host according to their mobility classes and hand off probabilities. The proposed model and scheme improved a connection blocking probability and a connection dropping probability efficiently.

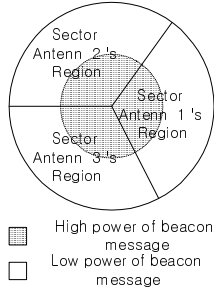
## 1 Introduction

Providing QoS in a wireless network is more difficult than in wired networks due to poor bandwidth in wireless networks and the mobility of mobile hosts which causes handoff. Handoff makes hardly servicing multimedia data. Additionally, it brings about drop of many packets. But it is difficult to estimate the handoff exactly. Therefore, many research papers used probabilistic methods to estimate handoff[1,2,3,4,5]. In pervious researches, all mobile hosts in a cell had the same handoff probability that they could move to a target neighbor cell. But, though the mobile hosts remain in a cell, each mobile host must have each other hand-off probability for the target neighbor cell by their movement patterns. So, we propose the differentiated handoff estimation model in this paper. In the differentiated handoff estimation model, if each mobile host comes through each other inner region in a cell, the each mobile host can have each other handoff probability for a target neighbor cell. The model is modeled with reducible Markov chain based on the inner regions in a cell.

Bandwidth is the most precious resource in the mobile networks and is changed rapidly due to mobility of mobile hosts. And, the mobility causes problems as a call blocking and a call dropping. So, an appropriate resource reservation scheme is required to reduce the call dropping and the call blocking. In resource reservation schemes for the mobile networks, when a mobile host came to a cell, the mobile host reserved proper bandwidth to neighbor cells[3,4,5,6].

In this paper, The dynamic resource reservation scheme can reserve differentiated bandwidth according to handoff probabilities of mobile hosts. And the dynamic resource reservation scheme updates the reserved bandwidth by changing mobility classes of the mobile hosts.

To classify inner regions in a cell for the differentiated handoff estimation model, cells are divided into three multiple inner regions by sector antennas and a power of beacon message. Figure 1 shows such architecture.



**Fig. 1.** Architecture for the differentiated handoff estimation model.

The rest of this paper is organized as follows: Section 2 reviews the related works regarding resource reservation on the mobile network; Section 3 presents the differentiated handoff estimation model using the reducible Markov chain; Section 4 proposes the dynamic resource reservation scheme. In section 5, we show the results of the performance analysis of our scheme. In conclusion, we present a summary of our research.

## 2 Related Works

In this section, we examine previous research done on call admission control and resource reservation on the mobile network. [3] proposes a resource reservation scheme based on the shadow cluster concept. The framework of the shadow cluster concept is a virtual message system for cellular system, which exchanges information concerning position or movement patterns with neighbor cells for reservation of bandwidth resources. [2] describes an active QoS hand off priority scheme which reduces the probability of call hand off failure in a mobile multimedia network with a micro/picocellular architecture. [4] presents two methods which use only local information to predict the resource demands and determine resource reservation levels for future handoff calls in a multimedia wireless IP networks. [5, 7, 8] are local methods in which a base station uses only local information to make resource reservation decisions.



$P_{ij}$  is the transition probability from state  $i$  to state  $j$ . If  $i, j \leq 3$ , then  $P_{ij}$  is the transition probability that a mobile host can move from the inner region  $i$  to the inner region  $j$ . Alternately, if  $i = j$ , then  $P_{ij}$  is the transition probability that a mobile host can still remain in the inner region  $i$ . And if  $i \leq 3, j \geq 4$ , then  $P_{ij}$  is the transition probability that a mobile host can move from the inner region  $i$  to the neighbor cell  $j$ . (3) is the transition probability matrix based on figure 2.

After a mobile host has came to a cell, the mobile host moves among inner regions until the mobile host will move to a neighbor cell.  $N_{ij}^{(n)}$  is the number of times that the mobile host visited in the inner region  $j$  until the mobile host moves  $n$  times. The mobile host came to the cell through the inner region  $i$ . In (4), we define a visited number of times as the indicated probability variables.

$$N_{ij}^{(n)} = \sum_{k=0}^n I_{ij}(k), I_{ij}(k) = \begin{cases} 1 & (X_k = j | X_0 = i) \\ 0 & \text{etc} \end{cases}, 1 \leq i, j \leq 3 \quad (4)$$

Because (4) begins at  $k = 0$ , the visited number of times includes a starting position. Therefore, (5) is formed.

$$E_{ij}^{(n)} = E[N_{ij}^{(n)}] = E\left[\sum_{k=0}^n I_{ij}(k)\right] = \sum_{k=0}^n E[I_{ij}(k)] = \sum_{k=0}^n P_{ij}^{(k)} \quad (5)$$

Because  $i, j$  are inner regions,  $P_{ij}^{(k)} = Q_{ij}^{(k)}$  using (2), we can show (6).

$$E_{ij}^{(n)} = \sum_{k=0}^n Q_{ij}^{(k)} \quad (6)$$

From (6)

$$Q_{ij}^{(0)} = \begin{cases} 1 & (i = j) \\ 0 & (i \neq j) \end{cases} \quad (7)$$

If the mobile host had came through the inner region  $i$ , then we regard that the mobile host first visited at  $i$ . (8) is expressed by a type of matrix using (6) and (7).

$$E^{(n)} = I + Q + Q^2 + \cdots + Q^n = I + Q(I + Q + \cdots + Q^{n-1}) = I + QE^{(n-1)} \quad (8)$$

If  $n$  is increased continuously, then the mobile host must move to one of the neighbor cells. Let  $\lim_{n \rightarrow \infty} E^{(n)} = \lim_{n \rightarrow \infty} E^{(n-1)} = E$  by (8). A matrix of the visited number of times for inner regions is (9).

$$E = I + QE = (I - Q)^{-1} \quad (9)$$

### 3.1 Estimation of Remaining Time

$T_i$  is the total number of times that the mobile host visited the inner regions till hand off.  $T_i$  is defined as (10).

$$T_i = \min\{n \geq 1; X_n = l | X_0 = i\}, i = 1, 2, 3 \text{ and } l = 4, 5, 6, 7, 8, 9 \quad (10)$$

We can calculate (11) by (4).

$$T_i = \sum_{k=0}^{m-1} \sum_{j=1}^3 I_{ij}(k) \quad (11)$$

So, we can calculate (12) by the expected value of the total visited number of times. In (12), after the mobile host moved  $m$  times, handoff is happened.

$$E(T_i) = E\left[\sum_{k=0}^{m-1} \sum_{j=1}^3 I_{ij}(k)\right] = \sum_{j=1}^3 E\left[\sum_{k=0}^{m-1} I_{ij}(k)\right] = \sum_{j=1}^3 E_{ij} \quad (12)$$

In (12),  $E_{ij}$  is the average number of times that the mobile host, after having started in the inner region  $i$ , visits the inner region  $j$  until handoff. So, we can show the average visited number of times  $E(T)$  by the addition of each row in  $E = (I - Q)^{-1}$ . Therefore, the number of times which the mobile host had visited inner regions is similar to time that the mobile host can remain in a cell.

### 3.2 Estimation of Handoff Probability

In this section, we estimate handoff probability in our proposed model. We define that the mobile hosts must move to neighbor cells before  $n$  movements.

$$A_{ij}^{(n)} = Pr(T \leq n; X_T = j | X_0 = i), \quad i = 1, 2, 3 \text{ and } j = 4, 5, 6, 7, 8, 9 \quad (13)$$

We calculate (14) using (2) and (8).

$$A^{(n)} = (I + Q + \dots + Q^{n-1})R = E^{(n-1)}R \quad (14)$$

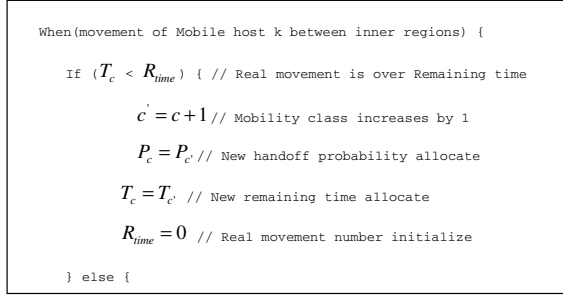
Therefore, we show a handoff probability matrix from (9) using  $n \rightarrow \infty$ .

$$A = ER = (I - Q)^{-1}R \quad (15)$$

### 3.3 Mobility Class

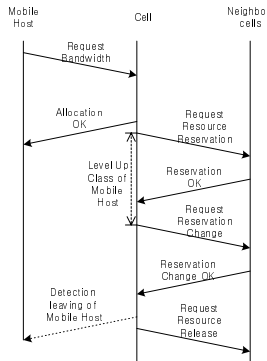
The differentiated handoff estimation model can dynamically control mobility class by change of remaining time. The mobility class shows how many a mobile host moves among inner regions.

In figure 3,  $T_c$  is remaining time of a mobile host  $k$  that has mobility class  $c$ . And,  $R_{time}$  is a real movement number of times that  $k$  has moved actually among inner regions in a current cell. If  $R_{time}$  exceeds remaining time, then the mobility class is increased by 1. The mobile host has a new hand off probability and a new remaining time according to the increased mobility class. Figure 3 shows dynamic management of mobility class.

**Fig. 3.** Dynamic mobility class management algorithm.

## 4 Dynamic Resource Reservation Scheme

Many traditional resource reservation schemes for the mobile networks were unable to change already reserved bandwidth. But, in our scheme, although a mobile host finished reservation of bandwidth successfully, the reserved bandwidth can be changeable by change of mobility class. Figure 4 shows flow of the dynamic resource reservation scheme. In figure 4, when the mobility class of a mobile host has been changed, the mobile host sends a reservation change message to neighbor cells. The semi-reservation scheme in [6] reserved part of requested bandwidth according to handoff probability of mobile hosts. And, our dynamic resource reservation scheme reserves part of or full requested bandwidth. But, the semi-reservation did not consider change of mobility class. Handoff probability in the semi-resource reservation scheme is simple ratio in historical record, but our scheme considers movement patterns of mobile hosts. And, the semi-reservation scheme can reserve part of requested bandwidth by only handoff probability ratio, but our scheme can adaptively reserve full or part of requested bandwidth according to bandwidth status of a cell. In the semi-reservation scheme, a rejected

**Fig. 4.** Flow of dynamic resource reservation.

resource reservation request may be accepted again by the changed bandwidth status in the neighbor cells. But, the amount of the reserved bandwidth could not be changed. Our dynamic resource reservation scheme can dynamically control already reserved amount of bandwidth by change of mobility class. The mobility classes in [6] could not be changed. However, the mobility class must be changeable because of the variable mobile environments. So we control dynamically the mobility class of mobile hosts by their remaining time.

After a mobile host has sent resource reservation message to neighbor cells, the neighbor cells tries the dynamic resource reservation.  $M = \{1, \dots, m\}$  is a set of neighbor cells and  $N = \{1, \dots, n\}$  is a set of mobile hosts current servicing from a neighbor cell  $m (m \in M)$ .  $B_{tot}$  is a maximum available bandwidth of the neighbor cell  $m$ .  $G = \{1, \dots, g\}$  is a set of servicing mobile hosts that belong to  $N$  and whose a current estimated departure time is larger than remaining time( $T_c$ ) of the requiring mobile host. The current estimated departure time is a value that subtract the estimated remaining time from a real movement time in the neighbor cell  $m$ .  $R = \{1, \dots, r\}$  is a set of reserved mobile hosts from the neighbor cell  $m$  and whose a current estimated arrival time is smaller than remaining time( $T_c$ ) of the requiring mobile host. The estimated arrival time is a value that subtract the estimated remaining time from a real movement time in their current servicing cell.

$$B_r = B_{tot} - \sum_{l=1}^g B_l - \sum_{k=1}^r B_k \quad (16)$$

$\sum_{l=1}^g B_l$  in (16) is the sum of bandwidth being use by mobile hosts belonging to  $G$  and  $\sum_{k=1}^r B_k$  is the sum of bandwidth reserved by requests belonging to  $R$ . Then  $B_r$  will be an available bandwidth of the cell  $m$  when the requiring mobile host comes to the cell  $m$ .

When a mobile host  $j$  requests a bandwidth reservation to the neighbor cell  $m$ , the dynamic resource reservation algorithm of the cell  $m$  is as following.

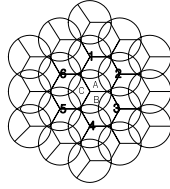
- Step 1.** If requested bandwidth( $b$ ) is smaller than  $B_r$ , then the request must be accepted.
- Step 2.** Otherwise,  $b$  is decreased by a decreasing rate  $d(b = b \times d, 0 < d < 1)$ .
- Step 3.** If decreased  $b$  is smaller than the  $B_r$ , the request must be accepted. Otherwise, if decreasing  $b$  is still larger than  $P_c \times b(original)$ , repeat step 2 and step 3 until decreasing  $b$  is smaller  $P_c \times b(original)$ . If the decreased  $b$  is still larger than  $B_r$ , go to step 4.
- Step 4.** The cell  $m$  sorts already reserved requests by descending according to their handoff probability.
- Step 5.** The cell  $m$  selects all reserved requests whose handoff probability is smaller than  $P_c$  among the sorted requests. The selected requests must belong to  $R$ .
- Step 6.** The cell  $m$  removes a reserved request that has the smallest handoff probability from the selected requests in order one by one until  $B_r$  becomes larger than original  $b$ .



**Step 7.** When all selected requests are removed, if the increased  $B_r$  doesn't become still smaller than original  $b$ , repeat step 2 until decreasing  $b$  is smaller than the  $B_r$ . If decreasing  $b$  is still larger than the  $B_r$ , the request must be rejected.

## 5 Numerical Results and Analysis

In this section, we present our simulation model and illustrate the simulation results. First, we show various handoff probabilities providing in the differentiated handoff estimation model. Second, we compare our dynamic resource reservation with the semi-reservation scheme in [6]. Each cell consists of three inner regions by sector antennas. We simulate the dynamic resource reservation and the semi-reservation on 19 cells. Figure 5 shows our simulation environment. Mobile hosts



**Fig. 5.** Simulation Environments.

are generated by the Poisson distribution. The velocity of the mobile host was allocated one of three cases (fast, medium and slow) randomly. A call duration time is 10 second basically and is additionally added to random time (maximum 600 second). When the mobile host generates in a cell or moves to a neighbor cell, the mobility class of the mobile host is 1 basically. And when the real movement time is larger than remaining time, the mobility class is increased by 1 until maximum 3. The mobility class in the semi-reservation is allocated randomly one of three classes (1, 2, 3).

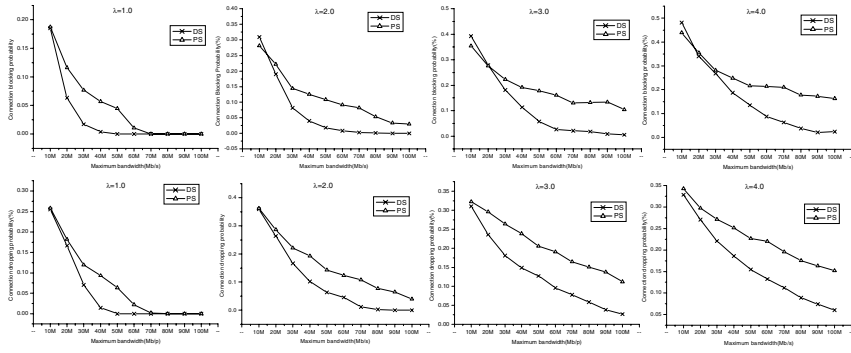
First, we present handoff probabilities in figure 5. A, B, and C are inner regions in a cell. 1, 2, 3, 4, 5 and 6 are neighbor cells. Table 1 shows differentiated handoff probabilities according to moving patterns and mobility class. Table 1 presents that various moving patterns of mobile hosts exist. Previous handoff estimation models without considering to moving patterns of mobile hosts estimated a same handoff probability for a target neighbor cell. But, our differentiated handoff estimation model considering to moving patterns of mobile hosts can estimate differentiated handoff probabilities to neighbor cells.

Table 1 shows handoff probabilities between inner regions and neighbor cells according to mobility classes. We observed about 3,500 handoffs in our simulation environment and produced the average handoff probability in the table 1.

Second, we present a performance of the dynamic resource reservation scheme. We use two important parameters for this simulation. The first parameter is

**Table 1.** Handoff Probabilities according to inner regions and mobility classes.

		Cell 1	Cell 2	Cell 3	Cell 4	Cell 5	Cell 6
Inner region A	Mobility Class 1	0.212809	0.227605	0.096844	0.174749	0.164705	0.123287
	Mobility Class 2	0.126516	0.216231	0.1702	0.14717	0.178035	0.160127
	Mobility Class 3	0.255474	0.133548	0.179453	0.113606	0.094259	0.223659
Inner region B	Mobility Class 1	0.155085	0.164724	0.137265	0.253364	0.165583	0.12398
	Mobility Class 2	0.108092	0.185556	0.205244	0.174689	0.171181	0.155237
	Mobility Class 3	0.232827	0.121714	0.201857	0.127396	0.093933	0.222284
Inner region C	Mobility Class 1	0.154545	0.161929	0.09608	0.176366	0.235462	0.175618
	Mobility Class 2	0.108994	0.18636	0.169515	0.146113	0.204295	0.184722
	Mobility Class 3	0.230744	0.120453	0.178229	0.112818	0.106077	0.251679

**Fig. 6.** Call blocking probability and call dropping probability.

the maximum available bandwidth in a cell. We change the maximum available bandwidth from 10Mbps to 100Mbps. And, the second parameter is the average generation rate of mobile hosts. The generation rate is 4 cases of 1.0, 2.0, 3.0 and 4.0. We show a call blocking probability and a call dropping probability in our simulation. In figure 6, DS is the dynamic resource reservation scheme proposed in this paper and PS is the semi-reservation scheme proposed in [6].  $\lambda$  is the generation rate. Figure 6 shows call blocking probabilities and call dropping probabilities according to  $\lambda$ .

When  $\lambda$  is 1.0 and the maximum available bandwidth is 70Mbps, call blocking probabilities both of DS and PS are 0. But when maximum bandwidth is 10Mbps in all  $\lambda$  cases, PS is slightly better than DS in case of call blocking probabilities. Because many mobile hosts generate in poor bandwidth, estimation of the hand off probability and remaining time are inaccurate. But, 10Mbps is low bandwidth to service multimedia data in a cell at future. And, figure 6 shows the call dropping probability for ongoing calls. We change 2 parameters(maximum bandwidth and generation rate) in simulation for call dropping probability, too. In figure 6, DS is better than PS. Increasing maximum bandwidth provide more correct estimation of the hand off probability and remaining time.

## 6 Conclusions

In this paper, we proposed the differentiated handoff estimation model based on inner regions in a cell and a dynamic resource reservation scheme. The differentiated handoff estimation model was modeled with the reducible Markov chain. And we provided the differentiated hand off probability and remaining time for the dynamic resource reservation scheme. The model provided differentiated handoff probability and remaining time according to mobility classes of mobile host. Our dynamic resource reservation scheme can dynamically control the mobility class of mobile host by remaining time. And, the dynamic resource reservation scheme utilizes the changing mobility class of mobile host immediately. We compared the dynamic resource reservation scheme with the semi-reservation scheme. In the results of our simulation, our dynamic resource reservation scheme is superior to the semi-reservation scheme. Our schemes improved the call blocking probability and the call dropping probability.

## References

1. C. Huang, "An analysis of CDMA 3G wireless communications standards," in Proc. IEEE VTC'99, Houston, TX, May 1999.
2. Wei Zhuang, Brahim Bensaoou, Kee Chaing Chua, "Adaptive Quality of Service Handoff Priority Scheme for Mobile Multimedia Networks," IEEE Transactions on Vehicular Technology, vol. 49, No. 2, March 2000.
3. David A. Levine, Ian F. Akyildiz, Mahmoud Naghshineh, "A Resource Estimation and Call Admission Algorithm for Wireless Multimedia Networks Using the shadow Cluster Concept," IEEE/ACM Transactions on Networking, Vol. 5, No. 1, February 1997.
4. Tao Zhang et al., "Local Predictive Resource Reservation for Handoff in Multimedia Wireless IP Networks," IEEE Journal on Selected Areas in Communications Vol. 19, No. 10, October 2001.
5. X. Luo, I. Thng, and W. Zhuang, "A dynamic pre-reservation scheme for handoffs with GoS guarantee in mobile networks," in proc. IEEE Int. Symp. Computers Communications, July 1999.
6. Geng-sheng Kuo, Po-Chang Ko and Min-Lian Kuo, "A Probabilistic Resource Estimation and Semi-Reservation Scheme for Flow-Oriented Multimedia Wireless Networks," IEEE Communications Magazine, February 2001.
7. L. Ortigoza-Guerrero and A. H. Aghvami, "A prioritized handoff dynamic channel allocation strategy for PCS," IEEE Transactions on Vehicular Technology, vol 48, July 1999.
8. S. Kim and T. F. Znati, "Adaptive handoff channel management schemes for cellular mobile communication systems," in Proc. ICC'99, 1999.
9. Ho-woo Lee, "Queueing theory," Sigma press, korea, 1999.

# Adaptive Generic Communications for Integrated Mobile and Internet Web-Services\*

Man-Ching Yuen, Leung Cheng, Pui-On Au, and Weijia Jia

Department of Computer Engineering and Information Technology  
City University of Hong Kong, 83 Tat Chee Avenue, Kowloon, Hong Kong  
{itconnie,itcheng,itpoau,itjia}@cityu.edu.hk

**Abstract.** This paper presents a novel Generic Communication Model (GCM) for integrated mobile and Internet web services. GCM is an adaptive protocol selection mechanism to ease the balance of transmission performance and communication interoperability in wired networks and wireless mobile networks. With GCM, end users can select a suitable protocol among different protocols for adapting different situations intelligently without any interruption and disconnection. In this way, it can provide seamless mobile communication by integrating both 3G telecom networks and IEEE 802.11 wireless networks through GCM which allow various client devices to communicate at their best acceptable QoS level through consistent and simple APIs. We present a preliminary prototype of GCM to illustrate its functionalities of communicating with heterogeneous mobile devices by using various communication protocols.

**Keywords:** Mobile Data Management, Web Services Technology.

## 1 Introduction

With the popularity of mobile devices, people can communicate easily at anytime anywhere. There are a large number of newly developed services for mobile devices. In current market, mobile devices including mobile phones and PDAs (Personal Digital Assistants) are able to support different platforms. It is necessary to maintain a centralized storage with an easy access interface, so it can provide QoS (Quality of Services) guaranteed services to any wired and wireless devices. To achieve this, we devise a platform called AnyServer [2] for supporting generic web and mobile services (such as mobile game services and mobile web contents services) to any wired and wireless devices. The objective of AnyServer platform is to provide any kind of services to any kind of client devices at anytime anywhere with the following philosophies:

---

\* The work is supported by Research Grant Council (RGC) Hong Kong, SAR China, under grant nos.: CityU 1055/00E and CityU 1039/02E and CityU Strategic grant nos. 7001446 and 7001587.

- Anytime and anywhere: AnyServer platform can be served as an information gateway and ensures web contents to be delivered smoothly from internet to mobile telecom network.
- Any kind of heterogeneous devices: AnyServer platform aims at integrating both Internet network and mobile telecom network, so heterogeneous mobile devices can interoperate with seamless communication and content can be displayed uniformly in devices of receivers.
- Any kind of services: AnyServer platform can serve as a service broker and provider to provide sets of services and common interfaces to clients on behalf of third party service providers through different network connection protocols.

To achieve the objective, a novel integrated wireless and Internet communication model for AnyServer platform called Generic Communication Model (GCM) has been proposed. By combining telecommunication paradigm with the unlimited borders of Internet (IP-network), AnyServer platform takes advantage of readily available mobile telecom network and powerful Internet to provide better services to those mobile users. As a result, AnyServer platform brings a comprehensive, cross-platform solution to people who enjoy seamless wireless communication.

Currently, a preliminary prototype of AnyServer platform has been implemented to illustrate the basic framework of the platform. The AnyServer prototype includes the mobile client module and the server module that inside the platform. With GCM in AnyServer platform, the mobile users using heterogeneous devices at the coverage area of wireless access points can have high speed connection to AnyServer platform.

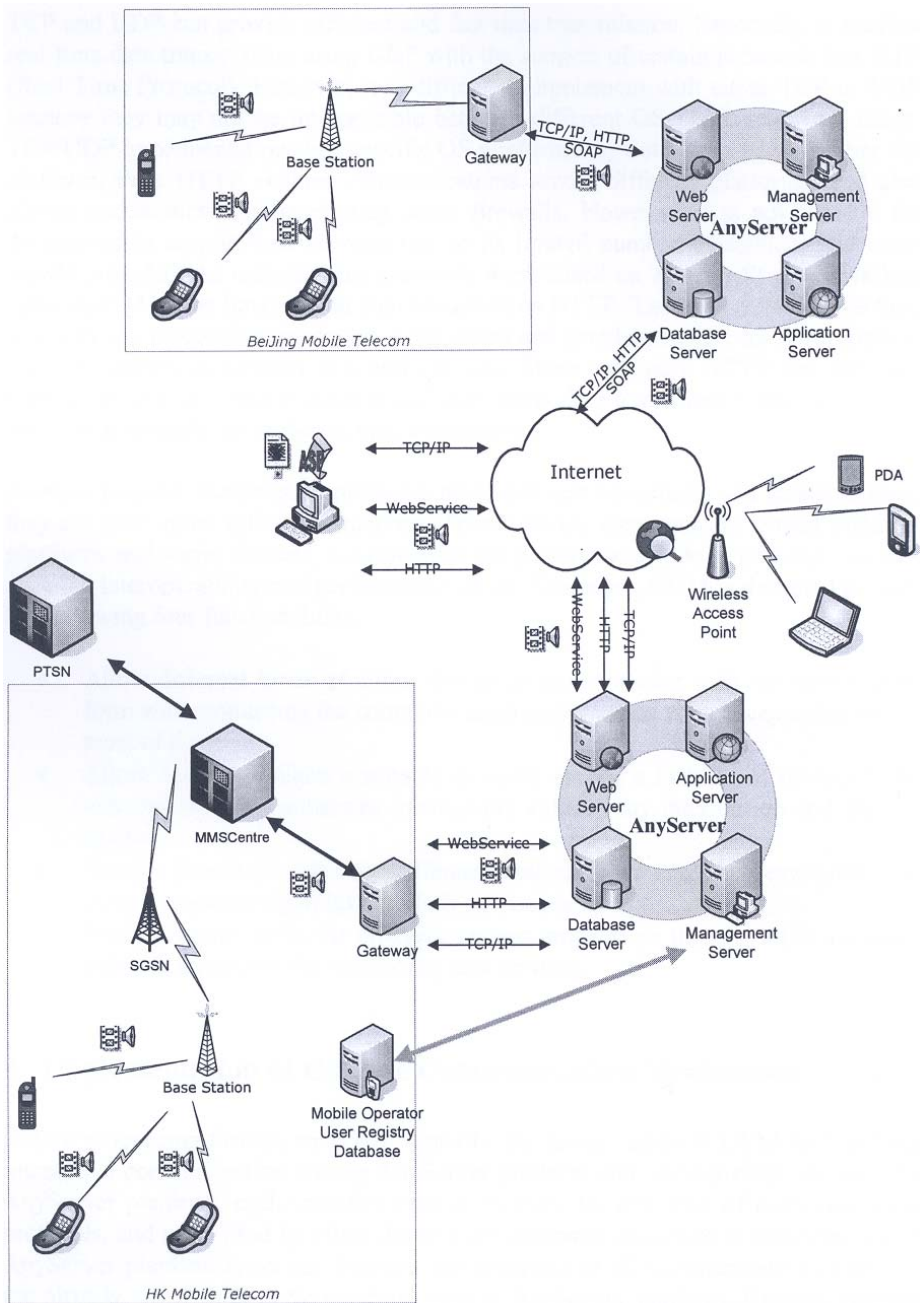
This paper presents the novel Generic Communication Model in AnyServer platform. Section 2 introduces the overview of GCM and its functionalities. Section 3 presents the design and implementation issues of GCM. Conclusion is drawn in Section 4.

## 2 Overview of Generic Communication Model

Generic Communication Model (GCM) in AnyServer platform is an adaptive protocol selection mechanism to ease the balance of transmission performance and communication interoperability among various clients and servers in wired networks and wireless mobile networks. Fig. 1 depicts the overall architecture of AnyServer platform with GCM. Based on features and popularities of different existing protocols, the transmission protocols used in GCM are classified as:

- TCP (Transmission Control Protocol) or UDP (User Datagram Protocol),
- Pure HTTP (HyperText Transfer Protocol) [7], and
- A web services protocol, SOAP (Simple Object Access Protocol).

TCP and UDP can provide efficient and fast data transmission. Especially, it enables real-time data transmission using UDP with the support of certain protocols like RTP (Real Time Protocol). However, it is difficult to implement with either TCP or UDP because they may not be interpretable between different OS platforms. As a



**Fig. 1.** Overall Architecture of AnyServer Platform with Generic Communication Model

result, TCP/UDP implementations in a specific OS platform may not be reused in another OS platform. Pure HTTP enables communications across different platforms, and also allows communications penetrating some firewalls. However, it is not flexible for developers to support new services due to its limited number of services and commands provided. As web services protocols work based on XML (eXtended Markup Language) [1], they have similar functionalities as HTTP. The only difference is that, web services protocols can provide a consistent and simple interface for developers to support connection services in a uniform way. Since both pure HTTP and web services protocols are used in application layer, their data transmission rate is low and may not be suitable for real-time data transmission.

As these popular transmission protocols have different advantages and disadvantages, they are used under different situations. To effectively communicate among different platforms and various devices, a mechanism for protocol selection is necessary to balance the interoperability and performance issues. Therefore, GCM is designed to have the following four functionalities:

- Allow different kinds of client devices to communicate with AnyServer platform with monitoring the communication performance at an acceptable level in most of the time.
- Allow clients to select a suitable protocol among a number of protocols for adapting different situations intelligently without any interruption and disconnection.
- Provide consistent APIs for different protocols thus ease the development of service modules regardless of platforms and devices.
- Provide simple APIs for different service modules so that the APIs are reusable and extensible for supporting new services.

### 3 Implementation of Generic Communication Mechanism

Before proceeding further, we briefly describe the design issues of GCM for handling interactive communication among AnyServer platform and various client devices. In AnyServer platform, each specified port is reserved for one kind of communication protocols, and ports used by client devices are randomly generated in real-time. Once AnyServer platform is set up, listeners and receivers of all communication protocols are already set at their corresponding ports at AnyServer platform. Besides, parsers supporting data formats of all transmission protocols are installed at AnyServer platform as well. Information of all the ports is given to client devices once our application programs are installed at the client devices. The design issues of GCM are categorized into two parts: connection establishment and transmission protocol selection. They are presented in Section 3.1 and Section 3.2 respectively. For the implementation issues of GCM, the TCP implementation and the web services implementation are described in Section 3.3 and Section 3.4 respectively. Note that we do not introduce the implementation of pure HTTP because it is similar to that of web services.

### 3.1 Connection Establishment

**Initialization of communication session between AnyServer platform and client device.** The communication session between AnyServer platform and client device can be initialized either by AnyServer platform or mobile client device depending on situations. In Fig. 2, a sender sends a request message to a receiver through AnyServer platform. In this case, the connection between the sender and AnyServer platform is initialized by the sender, while the connection between AnyServer platform and the receiver is initialized by AnyServer platform. By considering different protocols used by client devices, the way of communication between AnyServer platform and client devices is defined during the communication initialization stage.



Fig. 2. Connection between Sender and Receiver through AnyServer Platform

**Notification of both protocol and platform of all communication parties in AnyServer platform.** Whenever communication is initialized either by AnyServer platform or client devices, AnyServer platform has to know the type of both protocol and platform of client devices. It is because negotiations between AnyServer platform and client devices may be required for having services of the best performance. Besides, client devices have to detect the available transmission protocols provided by AnyServer platform during the communication initialization stage. It is useful for selecting the most adequate transmission protocol in later step. Note that installations of firewalls at AnyServer platform may cause client devices not able to detect the protocols provided by AnyServer platform during selection of protocol. For simplicity, in the first stage of implementation of GCM, we may only consider PDA as the client device which AnyServer platform already knows its platform and protocol, and adopt them without any selection.

### 3.2 Transmission Protocol Selection

**Definition of the common APIs (Application Program Interfaces) of supported transmission protocols.** To support most of the services provided by transmission protocols, we have to define the common APIs of the transmission protocols available in AnyServer platform. In this way, each service module has a set of APIs for providing its service to client devices where the APIs are able to support different protocols, platforms and client devices.

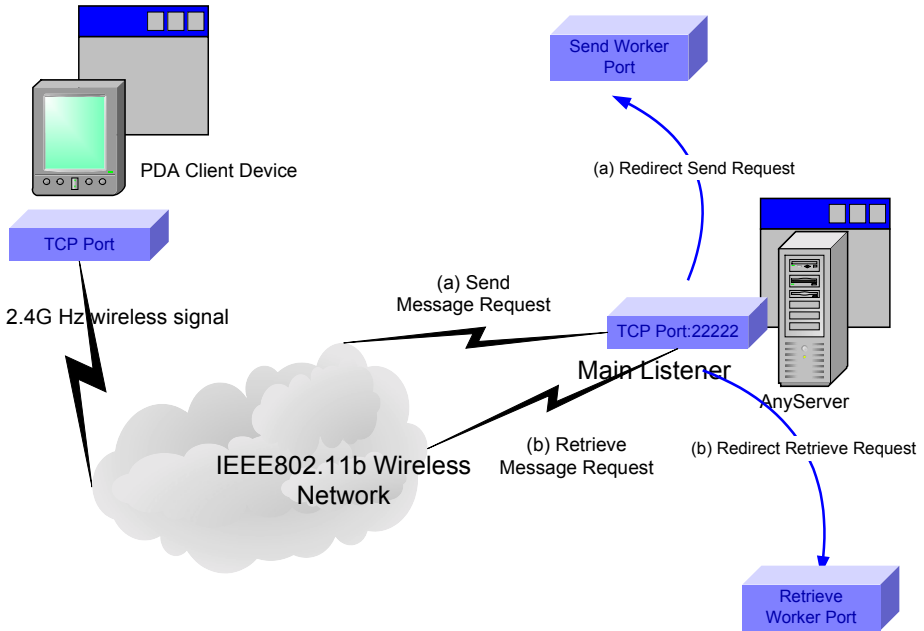
**Selection of the most suitable transmission protocol in different situations.** There exist a number of data type, such as non real-time text data and real-time video data. By considering the *performance* (user's point of view) and the *ease of development* (developer's point of view), different transmission protocols are suitable for different



natures of data transmission. To select the most suitable transmission protocol according to different situations, we have to find out the properties of data transmission way, the available protocols supported by client devices and the protocols provided by AnyServer platform. Once client devices have these information, they determine the most adequate transmission protocol for specific data. Then the AnyServer platform responds to the requests of client devices by providing an appropriate data transmission process using the most adequate protocol.

### 3.3 TCP Implementation

TCP (Transmission Control Protocol) serves as one of the efficient solutions in GCM because it can provide high speed data transmission. Fig. 3 illustrates the TCP implementation and the data transmission procedure when TCP is the selected protocol in GCM.



**Fig. 3.** TCP Implementation and its Data Transmission Procedure

To handle incoming requests, a *Main Listener* thread, which is independent of AnyServer platform, continuously listens on TCP port. At the current stage of our prototype design, it has not been extended to demonstrate the implementation of telecommunication networks. In this way, incoming requests are mainly initiated by mobile devices such as PDA through IEEE802.11b wireless networks [6] connecting to AnyServer platform. Main Listener distinguishes all incoming requests into two categories: *Send Message Request* and *Retrieve Message Request*. After that, based on categories of requests, Main Listener initiates a specific worker thread to handle

actual transmission of messages. *Send Worker Thread* and *Retrieve Worker thread* are created for handling Send Message Request and Retrieve Message Request respectively. This approach provides a very efficient and effective data transmission way particularly when incoming requests are bursty and in substantial amount. Fig. 4 shows the Algorithm applied in Main Listener of TCP.

```

Continue Loop
Listener.Accept  $\leftarrow$  New_Req
If New_Req.Req_Type = Send_req Then

    int  $ST_0 \in$  Sender_Pool()
    new S_Worker_Thread(  $ST_0$  )
    S_Worker_Thread.Listen;
    SendWorker_port  $\leftarrow$   $ST_0$ 
    Reply( SendWorker_port )

ElseIf New_Req.Req_Type = Recv_req Then

    int  $RT_0 \in$  Recv_Pool()
    new R_Worker_Thread(  $RT_0$  )
    R_Worker_Thread.Listen;
    RecvWorker_port  $\leftarrow$   $RT_0$ 
    Reply( RecvWorker_port )

End If

Listenr.ConnectClose()
End Loop

```

**Fig. 4.** Algorithm applied in Main Listener of TCP

When TCP is selected as the transmission protocol in GCM, Main Listener of TCP is initialized and starts listening on the TCP port continuously. Once a new request arrives, Main Listener accepts the request first. Then a field inside the request called Req\_Type is examined to determine whether it is a sending request or a retrieving request. In next step, Main Listener selects an available port number from the corresponding pool base, and initiates a worker thread that listens to the selected port to handle the detailed data transmission issues. After that, Main Listener replies to the client devices with the port number of the new worker thread, so the client devices further connect to the corresponding worker port. Once the data transmission is completed, Main Listener closes the connection and waits for the next incoming request.

### 3.4 Web Services Implementation

Web services is defined as a technology that can provide an interface for some functionalities to enable the accessibility from other programs across the web [5]. However, to achieve the design objectives of web services, several technologies associated with the same purposes have to be used together and they are listed as follows.

- Extensible Markup Language (XML) which is a standard data exchange format for cross-platform. XML schema for describing the structure of data.
- Simple Object Access Protocol (SOAP) [8] which is a standard data packaging method used for transmission over the Internet.
- In Web Services Description Language (WSDL) [4], Universal Description, Discovery and Integration (UDDI) which describes operations of web services and locates web services via capabilities or descriptions.

With these standards and protocols, web services can be consumed by anyone through Internet connection. In AnyServer platform, web services technology has been applied to increase the interoperability of AnyServer platform across heterogeneous operating systems. Microsoft C# [3] programming language is applied for the development of web services implementation in GCM. It not only automates building processes, but also self-generates the codes associated with above standards and protocols. It makes the entire development cycle to be reduced to a short period of time. In addition, because of web services having characteristics of passing through firewalls, it is adopted and implemented in GCM in AnyServer platform to support various connection protocols to end users without resistant from firewalls. The web methods functions like the way of a traditional Remote Procedure Call. It means that client devices can retrieve data from a remote server by simply calling the methods transparently through HTTP layer. In the following tables, some web services methods used in AnyServer platform are shown in details.

Table 1 describes the web services methods related to user profile management issues. In the user profile related web methods, end users can manage their detailed user information such as phone book entries and user preferences.

**Table 1.** Web Services Methods of User Profile Manipulation

NAME OF METHODS	DESCRIPTION OF METHODS
GetUser (void)	Get the user object from AnyServer platform.
SetPassword (string oldPassword, string newPassword)	Set or change the password of the user account.
UpdateUser (User user)	Update the user information by modifying the values of properties in the input User object.
UpdatePreference (Preference preference)	Update the preferences of the login user.
DelPhoneBookEntry (ulong phoneNumId)	Delete the phone book entry from the phone book of the user by giving the phone number.
UpdatePhoneBookEntry (PhoneBookEntry phoneBookEntry)	Update the phone book entry of the phone book of the login user.
AddPhoneEntry (PhoneBookEntry phoneBookEntry)	Add a phone book entry into the phone book of the login user.
Login (string userName, string password)	Authenticate the login user in AnyServer platform.
Logout (void)	Logout and remove any user login states associated with the user.
GetPhoneBook (void)	Get the phone book of the login user.
AddUser (User user)	Add a new user in AnyServer platform using the input User object.

In Table 2, it describes the web services methods for displaying node manipulation issues. By using these web methods, when users change the displayed nodes in their mobile client devices, the associations among the nodes are recorded in AnySer platform.

**Table 2.** Web Services Methods of Node Manipulation

NAME OF METHODS	DESCRIPTION OF METHODS
NewDirNode (string path, string name)	Create and add a DirNode under the given path using the given name.
DelDirNode (ulong nodeId)	Delete the DirNode and all its child nodes by giving the nodeId.
GetDirNode (string path, uint numOfLevel)	Get sub-directories and files under the given path.
DelFileNode (ulong nodeId)	Delete the node and its associated objects by giving nodeId.
GetFileNode (void)	Get the content files associated with the tree nodes.

Table 3 describes the web services methods for handling text message transmission. To limit the data volume transmitted in wireless networks, only the message headers are retrieved firstly to client devices through method GetAllMessageHeaders (void). After the user selects a particular message entry, the content of the selected message is transmitted to the client device through method GetMessageBody (ulong id, bool isInfoId). This approach can reduce the data volume transmitted as low as possible. Since mobile services providers charge end users based on the data volume transmitted, this approach can largely save the cost of data transmission for end users.

**Table 3.** Web Services Methods of Text Message Transmission

NAME OF METHODS	DESCRIPTION OF METHODS
GetAllMessageHeaders(void)	Get the message headers of all message entries.
GetMessage (ulong infoId)	Get a particular message including all the parts, using the given message ID.
GetMessageHeader (ulong infoId)	Get the message header of a particular message using the given message ID.
GetUnreadMessageHeaders (void)	Get the message headers of all the unread messages.
GetMessageBody (ulong id, bool isInfoId)	Get the message body of a particular message using the given message ID.

## 4 Conclusion

In this paper, a heuristic Generic Communication Model (GCM) for an integrated wireless platform, AnyServer platform, has been illustrated. The functionalities of GCM for communicating with heterogeneous mobile devices under different situations are given. Besides, the preliminary TCP and Web Services implementations of GCM are also presented. Since the entire AnyServer platform is still evolving, the

future development will consider providing a concrete solution in seamless wireless communication through integrating both mobile telecommunication networks and IEEE802.11b wireless networks by using Session Initialization Protocol (SIP) [9]. Since SIP identifies an incoming connection request as a session unit rather than handling a request on the application level of message, it is particularly suitable in various heterogeneous natures of networks.

## References

1. S. Hollenbeck, M. Rose, L. Masinter, Guidelines for the Use of Extensible Markup Language (XML) within IETF Protocols, RFC3470, Mar 2002
2. AnyServer, <http://anyserver.cityu.edu.hk>
3. Microsoft C# programming language, <http://msdn.microsoft.com/vcsharp/>
4. Web Services Description Language (WSDL), <http://www.w3.org/TR/wsdl>
5. Joseph Bustos and Karli Watson., Beginning .NET Web Services using C#, Wrox, 2002
6. Wireless LAN Alliance, The IEEE 802.11 Wireless LAN Standard, referred Oct 09th 1999  
<http://www.wlana.com/intro/standard/index.html>
7. T. Berners-Lee, R. Fielding, H. Frystyk, Hypertext Transfer Protocol -- HTTP/1.0 , RFC1945, May 1996
8. Simple Object Access Protocol (SOAP), <http://www.w3.org/TR/SOAP/>
9. J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, E. Schooler, Session Initiation Protocol ,RFC3261, June 2002

# Spatio-temporal Database with Multi-granularities

Sheng-sheng Wang and Da-you Liu

College of Computer Science and Technology,  
Key Laboratory of Symbolic Computing and Knowledge Engineering of Ministry of Education,  
Jilin University, 130012 Changchun, China  
wang\_sheng\_sheng@163.com, dyliu@mail.jlu.edu.cn

**Abstract.** Spatio-temporal information process grows up quickly in these years. Although uncertain and multi-granularities is the common features of spatio-temporal data, those problems were not well solved yet. A new spatio-temporal granularity representation method which supports multiple granularities and uncertain spatio-temporal objects is put forward. Then this method is applied to a spatio-temporal database ASTD. By supporting multiple granularities and approximate region, ASTD can process the multiple level data together, perform the query in variant precision and handle uncertainty. Compared with similar systems, ASTD is more powerful in multiple granularities, uncertainty, object type and granularity operations.

## 1 Introduction

Spatial (or geographical) information process becomes more and more important now, so does the spatio-temporal information since spatial and temporal information are inherently interrelated. Langran identified the need to describe spatial change over time and examined the design of temporal GIS in 1992<sup>[1]</sup>. Abraham reviewed the research of spatio-temporal database in 1999<sup>[2]</sup>. Hornsby and Egenhofer<sup>[3]</sup> presented an approach to represent and model geographic entities in time in 2000. In 2001, Peuquet<sup>[4]</sup> gave a detailed overview on today's issues in spatio-temporal representation. Many spatio-temporal data model have been proposed. But some critical problems such as uncertainty are still open. Multi-granularities is also a critical problem. Spatio-temporal Reasoning (STR for short) has wide variety of potential applications in AI and other application fields. Within the researches of STR, many efforts have been made in spatio-temporal (or spatial or temporal) granularity and uncertainty. From 1998 to 2001, Bettini proposed the Calendar Algebra which formally expressed the temporal granularity, and applied it to temporal database, temporal CSP and temporal data mining<sup>[5,6]</sup>. Based on Calendar Algebra and automata, Lago automatically named the time granule in 2001<sup>[7]</sup>. Bittner proposed the Rough Location method to represent approximate spatial region<sup>[8]</sup> or temporal interval<sup>[9]</sup> in 2000. The Rough Location is a rough set theory based spatial or temporal granularity theory. In 2003, Stell investigated the qualitative extents for spatio-temporal granularity<sup>[10]</sup>. Hornsby and Egenhofer proposed the multi-granularity method of moving object database (MOD) which is a simplified spatio-temporal database in 2002<sup>[11]</sup>.

Unfortunately no mature theory of spatio-temporal granularity is proposed yet. Bittner's<sup>[8,9]</sup> and Bettini's<sup>[5,6]</sup> theories are only suitable for single spatial or temporal aspect. Stell<sup>[10]</sup> described spatio-temporal granularity only by picture, no formal definitions or operations were given. Hornsby's work<sup>[11]</sup> on MOD can't be generalized to common spatio-temporal data (MOD only supports points object not regions which are known as ontology of objects<sup>[12]</sup>).

The overall goal of this paper is to propose a formal spatio-temporal granularity theory which inherently supports multiple granularities and uncertainty. Then applies it to a spatio-temporal database in a precision agriculture project.

## 2 Spatio-temporal Granularity

### Definition 1.

$S$  is the spatial domain which we study. There are  $m$  subparts of  $S$ , and they do not overlap each other:

$$SG(i) \subseteq S \quad (1 \leq i \leq m) \quad \text{and} \quad SG(i_1) \cap SG(i_2) = \emptyset \quad (1 \leq i_1, i_2 \leq m)$$

they form a **spatial granularity**  $SG$ , the  $SG(i)$  ( $1 \leq i \leq m$ ) is called a **spatial granule**.

In a finite spatial domain  $S$ , any spatial granularity  $SG$  has finite spatial granules, and they can be identified by the index. Index may be an integer:  $SG(1), SG(2), \dots$ ; or a text label, such as  $SG(\text{"Heilongjiang"})$ ,  $SG(\text{"Jilin"})$ ,  $\dots$ . For example, China map is the spatial domain  $S$ , each province is one spatial granule.

$SG_1$  and  $SG_2$  are two spatial granularities under the same spatial domain  $S$ .  $SG_1(i)$  and  $SG_2(j)$  are spatial granules of  $SG_1$  and  $SG_2$  respectively. If the space occupied by  $SG_1(i)$  is contained by the space occupied by  $SG_2(j)$ , then  $SG_1(i)$  is a finer granule than  $SG_2(j)$ , denoted by  $SG_1(i) <^* SG_2(j)$ . If every granule of  $SG_1$  is composed by some finer granules in  $SG_2$ :

$$\forall SG_1(i) \in SG_1 \rightarrow SG_1(i) = \bigcup_{SG_2(k) <^* SG_1(i)} SG_2(k) \quad \text{then } SG_2 \text{ is finer}$$

than  $SG_1$ , denoted by  $SG_2 = | SG_1$ . ( $SG, = |$ ) forms a partial order.

### Definition 2.

$T = [t_0, t_n]$  is the temporal domain. There are  $n$  continuous time intervals in domain  $T$ , named by  $TG(1), TG(2), \dots$ .  $TG(1) = [t_0, t'_1]$ ,  $TG(n) = [t'', t_n]$  and  $TG(j) = [t_1, t_2]$ ,  $TG(j+1) = [t_2, t_3]$ ,  $1 \leq j \leq n-1$ ,  $t_0 \leq t_1 < t_2 < t_3 \leq t_n$ .  $TG(j)$  is a **temporal granule**,  $TG$  is a **temporal granularity**.

The index of  $TG$  can also be integers or text labels.  $TG_1$  and  $TG_2$  are two temporal granularities under the same temporal domain  $S$ .

$TG_1(i) = [t_i^1, t_i^2]$ ,  $TG_2(j) = [t_j^1, t_j^2]$  are temporal granules of  $TG_1$  and  $TG_2$  respectively. If  $t_i^1 \geq t_j^1$  and  $t_i^2 \leq t_j^2$  then  $TG_1(i)$  is a finer granule than  $TG_2(j)$ , denoted by  $TG_1(i) <^* TG_2(j)$ . If every granule of  $TG_1$  is composed by some finer granules in  $TG_2$ :

$$\forall TG_1(i) \in TG_1 \rightarrow TG_1(i) = \bigcup_{TG_2(k) <^* TG_1(i)} TG_2(k)$$

then  $TG_2$  is finer than  $TG_1$ , denoted by  $TG_2 = | TG_1$ . ( $TG, = |$ ) forms a partial order.

For example, Domain T is one day, there are 3 granularities: hour, minute and second.  $TG_{second} = | TG_{minute} = | TG_{hour}$

Since the above conceptions of spatial granularity and temporal granularity are consistent and parallel, the definition of spatio-temporal granularity is formed by merging them together.

### Definition 3.

SG is a spatial granularity in spatial domain S, TG is a temporal granularity in temporal domain T. Given two granules  $SG(i) \in SG$  and  $TG(j) = [t_j^1, t_j^2] \in TG$ ,  $STG(i,j) = SG(i) \otimes TG(j)$  is a spatio-temporal granule,  $STG = SG \otimes TG$  is a spatio-temporal granularity.

If  $STG = SG \otimes TG$ , the projection from STG to SG and TG is denoted by :

$$\prod_S STG = SG, \prod_S STG(i, j) = SG(i),$$

$$\prod_T STG = TG, \prod_T STG(i, j) = TG(j)$$

If  $\prod_S STG_2 = | \prod_S STG_1$  and  $\prod_T STG_2 = | \prod_T STG_1$ , then  $STG_2$  is finer than  $STG_1$ , denoted by  $STG_2 = | STG_1$ . ( $STG, = |$ ) is a partial order.

## 3 Uncertain Spatio-temporal Representation

According to spatial ontology theory, a spatial object denoted by the region it occupied<sup>[12,13]</sup>. Due to the reasons of imprecision, vague conception or fuzzy data etc., most spatial objects are approximate<sup>[12,13,14]</sup>. That is to say, we can not exactly describe the detail of a object (or a region). Since the spatial granule is the minimal spatial unit which we can perceive, the approximate region can only be approximately represented by the spatial granule. According to Bittner's result<sup>[8]</sup>, the relationship between an approximate region  $R^*$  and one spatial granule could be: fully covers (F), partly overlaps(P) or not overlaps(N). So by enumerating all the granule with the value of {F,P,N},  $R^*$  can be qualitatively described.

By generalizing this method to spatio-temporal field, we get the representation of spatio-temporal object with spatio-temporal granularity. In a spatio-temporal granule, the spatial relation of approximate relation and spatial granule may various at different time. A grid is used to visually show dynamic state in a granule. Each row stands for one constant state (Fig. 1).

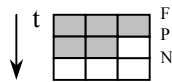


Fig. 1. Spatio-temporal granule



Without concerning the order (just enumerates all the appeared states), there are 7 JEPD (Jointly Exhaustive and Pairwise Disjoint) spatio-temporal relations for  $R^*$  and  $STG(i,j)$ . Note that all the grids in Fig.2 except F and N show only one possible instance. The relations are not strictly defined by the grid pictures which just help understanding.

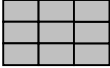
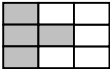
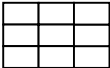

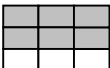
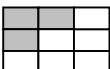
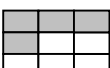
<b>F</b>		region always fully covers granule
<b>P</b>		region always partly overlaps granule
<b>N</b>		region never overlaps granule
<b>FP</b>		sometime fully covers and sometime partly overlaps
<b>FN</b>		sometime fully covers and sometime does not overlap
<b>PN</b>		sometime partly overlaps and sometime does not overlap
<b>FPN</b>		sometime fully covers, sometime partly overlaps sometime does not overlap

Fig. 2. Relation between  $STG(i)$  and  $R^*$

Under a certain and fixed spatial granularity, the basic spatial operations on approximate region is  $\cap_s$ ,  $\cup_s$  and  $\neg_s$ . Table 1 is the operation rules of spatial granule.  $P \cap_s P = \{P,N\}$  means that the result is uncertain, either P or N is possible.

Table 1. Operations on spatial relation

$\cap_s$	F	P	N
F	F	P	N
P		$P,N$	N
N			N

$\cup_s$	F	P	N
F	F	F	F
P		$P,F$	P
N			N

$\neg_s$	F	P	N
	N	P	F

**Table 2.** Rules of  $\cap_{st}$ 

$\cap_{st}$	F	P	N	FP	FN	PN	FPN
F	F	P	N	FP	FN	PN	FPN
P		P,P,N,N	N	P,PN	PN	PN,N	PN
N			N	N	N	N	N
FP				FP,P	FN,FPN,PN	PN,N	FN,PN,FPN
FN					FN, N	PN,N	FPN,PN,FP,N,N
PN						PN,N	PN,N
FPN							FPN,PN,FP,N,N

**Table 3.** Rules of  $\cup_{st}$ 

$\cup_{st}$	F	P	N	FP	FN	PN	FPN
F	F	F	F	F	F	F	F
P		P,F,FP	P	F,FP	FP	P,FP	FP
N			N	FP	FN	PN	FPN
FP				F,FP	F,FP	F,FP	F,FP
FN					F,FN	FN,FP,FPN	F,FN,FP,FPN
PN						P,PN,FP,FPN,FPN	FN,FP,FPN
FPN							F,FP,FPN,FPN

**Table 4.** Rules of  $-_{st}$ 

$-_{st}$	F	P	N	FP	FN	PN	FPN
	N	P	F	PN	FN	FP	FPN

Under a certain and fixed spatio-temporal granularity, the basic operations on approximate region is  $\cap_{st}$ ,  $\cup_{st}$  and  $-_{st}$ . To calculate this, Every possible time sequence combinations should be considered.

Take  $FN \cap_{st} FPN$  for example, from the following combinations:

$$(F \cap_s F) (F \cap_s P) (N \cap_s N) = (F)(P)(N) = FPN$$

$$(F \cap_s P) (F \cap_s N) (N \cap_s F) = (P)(N)(N) = PN$$

$$(F \cap_s F) (N \cap_s P) (N \cap_s N) = (F)(N)(N) = FN$$

$$(F \cap_s N) (N \cap_s F) (N \cap_s P) = (N)(N)(N) = N$$

we know  $\{FPN, PN, FN, N\} \subseteq (FN \cap_{st} FPN)$ . And  $\{F, P, FP\} \not\subseteq (FN \cap_{st} FPN)$  is deduced from  $(N \cap_s *) = N$ . So we have  $(FN \cap_{st} FPN) = \{FPN, PN, FN, N\}$ . Repeating the similar reasoning, we got Table 2, Table 3 and Table 4.

## 4 Granularity Conversion

In spatio-temporal database, GIS or other systems, data flows into system from multiple ways, so multiple granularities may exist simultaneously. In most cases, the data representations on different granularities can't be converted into each other. But coarsening granularity (transform information representation from finer granularity to coarse one) is feasible in theory. Coarsening information is also helpful in data min-

ing, database warehouse and hierarchy reasoning <sup>[12,15]</sup>. The method of transforming uncertain spatio-temporal representation form finer granularity to coarse one is discussed in this section.

To coarsen the representation of approximate region, firstly we coarsen the temporal granularity then the spatial one , or reverse the two steps.

**Definition 4. Coarsening temporal granularity**

Given two temporal granularities  $TG_1$  and  $TG_2$ , assume  $TG_2 = | TG_1$ .  $TG_1(j) \in TG_1$ ,  $TG_1(i) = \bigcup_{k \leq m} TG_2(i_k)$ ,  $TG_2(i_k) \in TG_2$ . Let  $R_1(j)$ ,  $R_2(j_k)$

be the spatio-temporal relations of  $TG_1(j)$  and  $TG_2(j_k)$  respectively. Then  $R_1(j) = R_2(j_1) \odot_t R_2(j_2) \odot_t \dots \odot_t R_2(j_m)$ , where operator  $\odot_t$  is to union the spatio-temporal relations by letters  $\{F, P, N\}$ .

**Table 5.** Coarsening temporal granularity

$\odot_t$	F	P	N	FP	FN	PN	FPN
F	F	FP	FN	FP	FN	FPN	FPN
P		P	PN	FP	FPN	PN	FPN
N			N	FPN	FN	PN	FPN
FP				FP	FPN	FPN	FPN
FN					FN	FPN	FPN
PN						PN	FPN
FPN							FPN

**Definition 5. Coarsening spatial granularity**

Given two spatial granularities  $SG_1$  and  $SG_2$ , assume  $SG_2 = | SG_1$ .  $SG_1(i) \in SG_1$ ,  $SG_1(i) = \bigcup_{k \leq m} SG_2(i_k)$ ,  $SG_2(i_k) \in SG_2$ . Let  $R_1(j)$ ,  $R_2(i_k)$  be the spatio-

temporal relations of  $SG_1(i)$  and  $SG_2(i_k)$  respectively . Then,  $R_1(i) = R_2(i_1) \odot_s R_2(i_2) \odot_s \dots \odot_s R_2(i_m)$ .

$\odot_s$  can't be calculated automatically , all the possible cases have to be considered. All the  $\odot_s$  are given in Table 6.

**Table 6.** Coarsening spatio-temporal granularity

$\odot_s$	F	P	N	FP	FN	PN	FPN
F	F	P	P	FP	FP	P	FP
P		P	P	P	P	P	P
N			N	P	PN	PN	PN
FP				FP,P	FP,P	P	FP,P
FN					P, FN, FP, PN, FPN	P, PN	P, FP, PN, FPN
PN						P, PN	P, PN
FPN							P, PN, FP, FPN

## 5 Spatio-temporal Database

The methods and theories discussed above were applied to Agrarian Spatio-Temporal Database (ASTD) which is a subproject of Integrated Development Platform of Intelligent Agriculture Information System(IDPIAIS) project supported by National “863” Project. ASTD is a spatio-temporal database manage system which can manage, query and analyze spatio-temporal agrarian information. Later the information would be send to other parts of IDPIAIS such as expert system for further reasoning. In this section, we focus on the ASTD of Nongan county (located in Jilin Province , China) , the spaito-temporal granularity theory had been applied to it.

In Nongan county, soil sampling is performed to gather and manage agrarian information every week from 1999 , four years saptio-temporal agrarian data was managed by ASTD. The soil sample data contains one hundred or so fields including nutrient concentrations (such as nitrogen, phosphor, kalium), meteorologic data(such as rainfall, air temperature, humidity ) ,organic matter, agrotype, breed, yield etc. . Each soil sample data was combined with time and location. Mobile GPS devices were used to locate the sample points.

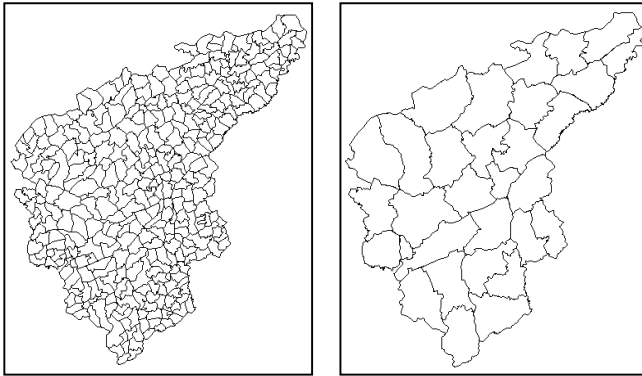


Fig. 3. Spatial Granularities

Depending on soil sample points, the land is divided into small polygons by voronoi graph. method. All the area within a voronoi polygon has the same attributes as the sample point in it. The voronoi graph forms a spatial granularity, and the time interval of sampling forms a temporal granularity. A spatio-temporal granule is formed by combining them. Besides sample, there are other three levels of spatial granularities in ASTD: Cun, Xiang and County(Fig. 3). They are all Chinese regionalism, ordered by gradual higher level. The relation of them is :  $SG_{sample} \supset SG_{cun} \supset SG_{xiang} \supset SG_{county}$ . As for temporal granularity, ASTD has four levels of granularities ( $TG_{week} \supset TG_{month} \supset TG_{quarter} \supset TG_{year}$ ). There are totally  $4 \times 4 = 16$  spatio-temporal granularities. All these spatio-temporal granularities form a Granularities Lattice. The most finer granularity is  $STG_{sample-week} = SG_{sample} \otimes TG_{week}$ .

As soil sampling is the data source. of ASTD, the voronoi graph regions (or say the granules of  $STG_{sample-week}$ ) is the smallest element of information. The soil sample data is associated with those regions as attributes. More coarser information can also

be get in ASTD, such as " Rainfall of Huajia in 2001". This kind of query need some coarse granularities. To get the attribute of coarse granule, we must merge the attributes of all the  $STG_{\text{sample-week}}$  granules within it. For the non-numeric type attributes such as *text*, all the different values must be united. For the numeric type attributes, the coarse value may be the sum , average , maximum, minimum,..., or weighted sum , weighted average,... of all the fine ones.

With the spatio-temporal granularities system of ASTD , approximate regions can be represented. Some spatial objects can be exactly described (such as rivers, factories, schools) while others can't (such as classifications of soil, pollution area ). By applying the theories and methods we mentioned early, all uncertain objects could be denoted with special STGs respectively. During its lifetime, snapshots of the approximate region were recorded.

In ASTD or other similar systems, besides uncertainty there is another critical problem. Too many granularities exist and users as well as requires are variant , so information is often recorded and retrieve with different granularities. Thus granularities conversion is an absolutely necessarily operation. But no up-to-date Temporal GIS or spatio-temporal database systems supports such operation.

By the methods discussed above, ASTD supports uncertain objects and the granularity of uncertain objects can be coarsened as required.

In ASTD , uncertain objects as well as attributes of multi-granularities can be queried by visual GUI and spatio-temporal query language. The syntax of query language is based on XML , technology details are introduced in another paper<sup>[16]</sup>.

GUI and query language can process such requests as:

"phosphor concentration of *Huajia Xiang* in 1999"

(attribute of granule  $STG_{\text{xiaing-year}}(\text{"Huajia"}, 1999)$  )

"rainfall of Sample Point 0268 in 2001.3"

(attribute of granule  $STG_{\text{sample-month}}(268, \text{"2001-3"})$  )

Uncertain information can be retrieve by querying the approximate region:

"which *Cun* were disserved by locust all through the third quarter of 2002"

(approximate region in  $STG_{\text{cun-quarter}}$  , spatial granules with the relation of F,P or FP )

"which *Cun* were whole disserved by locust in the third quarter of 2002"

(approximate region in  $STG_{\text{cun-quarter}}$  , spatial granules with the relation of F, FN, FP or FPN )

And some query need perform operations on two or more approximate region:

"which *Cun* are disserved by locust and aphid simultaneously in 2001"

( $\cap_{\text{st}}$  operation on granules in  $STG_{\text{cun-year}}$ )

ASTD is developed by WebSphere with J2EE standard . The console and user interface are all performed in web browser. The attributes are stored in ORACLE while spatial objects are manage by SDE(spatial data engine) of ORACLE. The spatio-temporal data was transformed through Internet by XML.

## 6 Conclusion

Granularity is an important subject of data representation. Although spatio-temporal information process grows up quickly in these years, its granularity and uncertain

question were not well solved. In this paper, we put forward a new spatio-temporal granularity representation which support multiple granularities and uncertain spatio-temporal objects. Finally, the theories and methods were applied to a spatio-temporal database ASTD. By supporting multiple granularities and approximate region, ASTD can process the multiple level data together, perform the query in variant precision and handle uncertainty. Compared ASTD with similar systems, it is more powerful in multiple granularities, uncertainty, object type (point is a special case of regions) and granularity operations such as coarsening. The theory discussed here can also be applied to spatio-temporal data mining, spatio-temporal CSP, spatio-temporal Logic etc.

## Acknowledgments

This research was supported by the project from NSFC of China (Grant No. 60373098) and the project from "863" Project(Grant No. 2003AA118020).

## References

1. Gail Langran, *Time in Geographic Information Systems*, Technical Issues in Geographic Information Systems. Taylor & Francis, 1992
2. Tamas Abraham AND John F. Roddick, *Survey of Spatio-Temporal Databases*, *GeoInformatica* 3:1, 61~99 (1999)
3. Hornsby, K. and Egenhofer, M. Identity-Based Change: A Foundation for Spatio-Temporal Knowledge Representation. *International Journal of Geographical Information Science* 14:3 (2000) 207-224
4. D.J. Peuquet, *Making Space for Time: Issues in Space-Time Data Representation*. *GeoInformatica*, 5:1 (2001) 11-32
5. C. Bettini, X. S. Wang, and S. Jajodia. A general framework for time granularity and its application to temporal reasoning. *Annals of Mathematics and Artificial Intelligence*, 22(1-2):29-58, 1998
6. C. Bettini and R. De Sibi. Symbolic Representation of User-Defined Time Granularities. *Annals of Mathematics and Artificial Intelligence*, 30(1-4), 2001
7. Ugo Dal Lago and Angelo Montanari, *Calendars, Time Granularities, and Automata*, C.S. Jensen et al. (Eds.): *SSTD 2001*, LNCS 2121, pp. 279-298, 2001
8. Thomas Bittner, T & Stell, J., *Rough sets in Approximate spatial reasoning*, *Proceedings of RSCTC'2000*, Berlin-Heidelberg Springer-Verlag, 2000:145-156
9. Thomas Bittner, *Approximate Qualitative Temporal Reasoning*, *AAAI 2000*:200-215
10. John G. Stell, *Qualitative Extents for Spatio-Temporal Granularity Spatial Cognition and Computation*, 2003, vol 3 no 2 & 3, pp119-136
11. Hornsby, K. and M. Egenhofer, *Modeling moving objects over multiple granularities*, *Special issue on Spatial and Temporal Granularity*, *Annals of Mathematics and Artificial Intelligence*, Kluwer Academic Press., 2002
12. A.G.Cohn and S.M. Hazarika, *Qualitative Spatial Representation and Reasoning: An Overview*, *Fundamental Informatics*, 2001, 46 (1-2):1-29
13. WANG Sheng-sheng, LIU da-you, YANG Bo. *Multi-dimensional Qualitative Spatial Query Language MQS-SQL*, *Acta Electronica Sinica*, 2002, 12A, 1995-1999

14. WANG Sheng-sheng , LIU Da-you , HU he. Approximate Spatial Relation Algebra ASRA and Application. Journal of Image and Graphics , 2003(8): 946~950
15. WANG Sheng-Sheng, LIU Da-You, XIE Qi. Integrating Multi-aspects Information for Qualitative Spatial Reasoning and Application, Journal of Software, 2003, 14(11): 1857~1862
16. WANG Sheng-Sheng, LIU Da-You. Spatio-temporal Query Language Based on Spatio-temporal Reasoning and XML. Computer Applications. 24(2),2004:118~121

# A Simple but Effective Dynamic Materialized View Caching

Chi-Hon Choi<sup>1</sup>, Jeffrey Xu Yu<sup>1</sup>, and Hongjun Lu<sup>2</sup>

<sup>1</sup> The Chinese University of Hong Kong, Hong Kong, China  
{chchoi,yu}@se.cuhk.edu.hk

<sup>2</sup> The Hong Kong University of Science and Technology, Hong Kong, China  
luhj@cs.ust.hk

**Abstract.** Dynamic materialized view management problem has recently received considerable attention. The problem is how to dynamically maintain some results of online analytical processing (OLAP) queries for the purpose of maximizing the possibility to answer subsequence ad-hoc OLAP queries as well as minimizing the total query processing cost. However, the existing dynamic view management approaches have their limit to support OLAP queries. In this paper, we propose a dynamic materialized view management approach with a new hit prediction, which can predict the users' interest in an ongoing fashion and adapt to changes if necessary. Extensive performance studies show that our approach achieves a high cache hit ratio and speeds up the response time or throughout for different kinds of OLAP queries.

## 1 Introduction

Data warehouse and OLAP systems have been widely used for business data analysis. Pre-computing OLAP queries (materialized view) becomes a key issue to achieve high query performance in data warehouses. Static materialized view management selects materialized views based on historical data, and attempts to use those materialized views for later reuse [1, 5–8, 12, 14]. However, users' interest may change over time and their OLAP queries may change as time proceeds accordingly. As a result, their ad-hoc query patterns, which are unknown in advance, are difficult to predict and make the selected materialized views quickly become outdated.

In order to effectively and efficiently handle unpredictable ad-hoc query patterns and minimize the total OLAP query processing costs, dynamic view management has been received significantly attention [3, 4, 9, 11, 13, 15]. The dynamic view management problem is to maintain the materialized views dynamically in the cache to retain up-to-date materialized views, and fully utilize the existing materialized views for maximizing the possibility to answer subsequence ad-hoc OLAP queries under a given limited space. The strategy of using cached results to process a query is effective when the query stream exhibits a high degree of locality.



The most important issue that has impacts on the performance of the dynamic materialized view management is the cache replacement policy. A cache replacement policy determines which view should be retained in the cache under a limited cache space. As time passes, some new views may need to be additionally materialized while the existing materialized views may not be able to answer the new incoming OLAP queries. Due to the limited cache size, some old materialized views are replaced by the new materialized views. In general, a replacement policy is a function which returns a value for a cached view. When space must be made available in the cache, the view(s) with the smallest values are selected as victims to release memory space.

In this paper, we extend the work in [2] and study a simple but effective caching mechanism that can support OLAP queries without predicates as well as those with predicates. Our dynamic materialized view management can dynamically detect the change of the users' interest and adapt the changes. Therefore, it results in retaining the up-to-date and most beneficial materialized views in the cache. Our new approach can reach a higher hit ratio and minimize the total query processing cost.

## 2 Preliminaries and Problem Definition

An  $M$ -Multidimensional Database (MDDb) is a collection of relations,  $D_1, \dots, D_M, F$ , where  $D_i$  is a dimension table and  $F$  is a fact table as described in [1, 10] (also known as star-schema). Each dimension table has a key attribute. The fact table keeps the foreign keys of the  $M$  dimension tables, with additional measures. Given an  $M$ -dimensional MDDb, a *dependent lattice* can be defined as  $(L, \preceq)$  [8], with a set of OLAP queries (group-by queries)  $L$ , and a dependence relation  $\preceq$  (derived-from, be-computed-from). Given two queries  $q_i$  and  $q_j$ . We say  $q_i$  is dependent on  $q_j$ , ( $q_i \preceq q_j$ ), if  $q_i$  can be answered using the results of  $q_j$ . A dependent lattice can be represented as a directed acyclic graph,  $G = (V, E)$ . Here  $V$  represents the set of OLAP queries, as vertices. We use  $V(G)$  and  $E(G)$  for the set of vertices and the set of edges of a graph  $G$ . An edge,  $v_i \rightarrow v_j$ , exists in  $E$ , if and only if  $v_j \preceq v_i$  and  $\nexists v_k (v_j \preceq v_k \wedge v_k \preceq v_i)$ , for  $v_i \neq v_k \neq v_j$ . A *datacube* is a dependent lattice with  $2^M$  vertices, for an  $M$ -dimensional MDDb, where each vertex represents a group-by.

The dynamic materialized view management is defined as how to maintain some results of OLAP queries in a limited space, in order to maximize the possibility to answer incoming OLAP queries in runtime.

## 3 Related Work

In this section, we outline three approaches [2, 4, 11] for dynamic view management, **Chunked-file**, **DynaMat**, and **Predicated-based**. The first two approaches cache the granularity of data as either chunk or fragment to support OLAP queries while the last one uses a dynamic predicate-based partitioning approach. Then, we address our concerns with the three approaches.

- **Chunked-file** [4]: Deshpande et al proposed a **Chunked-file** which divides the multidimensional query space into uniform chunks. **Chunked-file** uses multi-dimensional arrays to store chunks, where a chunk, ranging at any level in the hierarchy, is proportional to the number of distinct values in the corresponding dimension at that level. When a query is issued, the system determines whether the query can be computed in three ways: (i) directly from the previous query results which are stored in the cache or a dedicate disk storage; or (ii) partially from the cached query results and partially from the base table; (iii) entirely from the base table.
- **DynaMat**[11]: Kotidis and Roussopoulos proposed a system, called **DynaMat**, which constantly monitors incoming queries and dynamically materializes information at multiple levels of granularity in the form of fragments. However, **DynaMat** puts some restrictions on the query patterns. **DynaMat** can only efficiently support the following three types of multidimensional range queries: (i) select a full range, that is the value is between the minimum value and maximum value of a particular dimension; (ii) a single value; and (iii) an empty range which means the dimension is not in the present in the query.
- **Predicated-based** [2]: Choi, et al proposed a dynamic predicate-based materialized view management approach for selecting and partitioning materialized views to reduce the query response time and maintenance cost. Its aims are to minimize the space consumption in a relational database system and fine-tunes the materialized views as much as possible to serve the future queries.

Despite the efficiency of the above these three approaches as reported in [2, 4, 11]. Some observations can be made below.

- The majority of data warehouses are built on top of existing relational database systems. The chunk-based caching approach, **Chunked-file**, is not directly applicable, and cannot be widely used. The efficiency of **Chunked-file** is at the expense of space consumption. **Chunked-file** may result in a huge number of empty chunks, in particular, when data is sparse. Consequently, the query processing cost using a chunked file can be high, because it needs to access a large number of chunks. The high space consumption is somehow against its goal to maximize the possibility of supporting more queries using a limited space. In addition, it is difficult for **Chunked-file** to determine the optimal chunk sizes.
- **DynaMat** partitions data into fragments. Only some types of OLAP queries can be supported efficiently. The reason that **DynaMat** cannot support a wide range of OLAP query types is due to the fact that the cost of finding a set of fragments to answer an OLAP query can be high itself. In addition, it is difficult for **DynaMat** to select a dimension(s) as the basis to partition data.
- **Predicate-based** caches OLAP queries in a relational multidimensional database and uses user predicates to partition views into horizontal fragments such that minimizing the total query processing cost. But, it can only support OLAP queries efficiently in the presence of user predicates.

## 4 A Simple but Effective Dynamic Materialized View Caching

We have developed a ROLAP-based dynamic materialized view management system, which can be easily adopted on top of relational database management systems. Our system consists of four main components: a query analyst, a view advisor, an adaptive learning cache manager, and a view cache. These components are described as follows, in brief.

- The query analyst parses an incoming query and converts the necessary information into internal data structures that will be used in other components.
- The view cache is the information repository that stores the materialized views.
- The view advisor selects the best view among the set of view candidates in the view cache to answer the query based on the derived-from relationship. If there are no qualified view, the query will be answered by the base tables. In brief, for an incoming query  $q$ , the view advisor will determine the set of materialized views that can answer  $q$  by checking if there are views in the view cache such that  $q \preceq v_i$  on the following conditions: i) all attributes in  $q$  must also be in  $v_i$ , ii) the selection conditions specified in  $q$  implies that it is a subset of  $v_i$ , and iii) the aggregate functions used in the two queries are the same. In order to eliminate the overhead, our view advisor only selects one view to answer a given query. It is because, in general, the cost of selecting best multiple views to answer  $q$  is high, and is exponential in terms of the number of views available. When there is no single view that can answer the query, the view advisor simply uses the base tables to answer the query.
- The adaptive learning cache manager applies a replacement policy to add some beneficial materialized views to replace outdated materialized views in the view cache, when the view cache becomes full. The replacement policy applies a value function to each of the cached views. The views with the lowest values are chosen as replacement victims for utilizing cache space.

We process OLAP queries in a unit of batch. By doing so, even a query cannot be answered by the existing materialized views in the view cache, it may be answered by a query result which appears in the same batch. Hence, within a batch, we assign a priority to each query according to their ability to serve other queries within the same batch. In other words, a query which can answer more queries in the same batch will be answered first. As a result, its query result is used to answer other queries. Batches offer an advantage to detect users query pattern changed.

In the following, we will discuss adaptive learning cache manager in detail. The key issue is replacement policies. In general, we consider such a policy as a function  $G$  (for goodness). For an existing view,  $V_i$ , in the view cache,  $G(V_i)$  returns a value to indicate how important it is for us to maintain it in the view cache for later reuse. When space must be made available in the view cache,

the views with the lowest values are chosen as replacement victims. It is worth of noting that the two commonly used caching techniques, Least Recently Used and Most Recently Used, are based on the assumption of that views that have been referenced recently are likely to be referenced again in the near future. They are not suitable for handling OLAP queries, because OLAP queries are group-by queries in nature, and the derived-from relationships cannot be easily managed by them.

In [11], Kotidis and Roussopoulos studied several functions including a Small-Fragment-First (**SFF**), and Small-Penalty-First (**SPF**). The intuition behind **SFF** is that larger views are more likely to be hit by a future query. A larger view implies fewer number of views stored in the view cache. Fewer number of views will reduce the overheads to find better views and replace views. But, **SFF** does not consider query frequency. If a larger partition is not a hot access region, it may waste the space and slow down the whole query response time as other hot queries have to access the base table. The goodness of a view  $V_i$  is measured by its size below.

$$G(V_i) = |V_i| \quad (1)$$

**SPF** considers query frequency, query processing time and size of a materialized view. The goodness of  $V_i$  is defined as

$$G(V_i) = \frac{f_{V_i} \times qcost(V_i)}{|V_i|} \quad (2)$$

where  $f_{V_i}$  is the query frequency of accessing  $V_i$ ,  $qcost(V_i)$  is the cost of recomputing  $V_i$  if it is removed, and  $|V_i|$  is the view size.

We observe that **SFF** and **SPF** cannot detect the changing of users' query patterns, and propose a new function, **HP** (Hit-Prediction) that can detect and adapt users' interest changes from the previous query patterns. Here, suppose  $V_i$  is a view that can be used to answer a new OLAP query.

$$G(V_i) = (|T| - |V_i|) \times (L_i + P_i) \quad (3)$$

where  $|T|$  is the table size of the base table in the data warehouse and  $|V_i|$  is the table size of the materialized view  $V_i$ . The difference of  $|T| - |V_i|$  represents the cost saving of keeping  $V_i$  in the view cache. Also, we consider two values,  $L_i$  and  $P_i$ , where  $L_i$  is a max local hit of  $V_i$  in the current batch of queries, and  $P_i$  is a predicted hit for the materialized view  $V_i$  in the future. The max local hit  $L_i$  is the total number of times a view  $V_i$  can be used to answer queries in the current batch. The max local hit is used to test whether the pattern of using  $V_i$  is changed or not. If the pattern of using  $V_i$  is changed, then we lower its  $P_i$  value, because we may not use it in a long term. If the pattern of using  $V_i$  is very similar with the past patterns, we give  $P_i$  a higher value, in order to keep it long. We attempt to maximize the cache hit to improve the total query processing time in long term.

Consider a view  $V_i$  that can be used to answer OLAP queries in the current batch. We explain how we calculate  $P_i$  for  $V_i$  as shown in Algorithm 1. When

---

**Algorithm 1** Hit Prediction.

---

**Input:** a view  $V_i$ ;**Output:** a predict hit  $P_i$ .

```

1: begin
2: obtain the max hit,  $L_i$ , for  $V_i$  in the current batch of queries;
3: if  $L_i = 0$  then
4:    $P_i \leftarrow 0$ ;
5: else if  $V_i$  is first hit in the current batch then
6:    $P_i \leftarrow L_i$ ;
7: else
8:   let  $M_i$  and  $Std_i$  be the mean and standard deviation of hits for  $V_i$  in all batches;
9:   if  $L_i$  is in the range of  $[M_i - Std_i, M_i + Std_i]$  then
10:     $P_i \leftarrow L_i$ ;
11:   else
12:     $P_i \leftarrow \gamma \times \frac{L_i + M_i}{2}$  where  $0 < \gamma \leq 1$ ;
13:   end if
14: end if
15: return  $P_i$ ;

```

---

$L_i = 0$ , that is, there are no queries in the current batch that can utilize  $V_i$  based on the derived-from relationship,  $P_i$  is simply set to zero. If it is the first time that the materialized view  $V_i$  is possibly hit,  $P_i$  is set to the max local hit  $L_i$ , because it is considered as a bursting pattern. Otherwise, we calculate  $P_i$  for  $V_i$  as follows. We introduce a statistical quality control to control  $P_i$ . The idea is to determine whether the usage of  $V_i$  is changed or not, by testing whether or not the max local hit  $L_i$  is within an acceptable range. In other words, if  $L_i$  is in a range of  $[M_i - Std_i, M_i + Std_i]$  then the pattern of using  $V_i$  is not changed, where  $M_i$  and  $Std_i$  are the mean and standard deviation of the view  $V_i$  being hit up to date. If there is no changed,  $P_i$  is set to  $L_i$ . It avoids noise and impulse and reach a nature fluctuation. If there is a possible change, we balance the max local hit  $L_i$  and  $M_i$ , and set  $P_i$  to  $\gamma \times (L_i + M_i)/2$  where  $0 < \gamma \leq 1$ .<sup>1</sup> We will discuss its efficiency in our experimental studies.

## 5 A Performance Study

We conducted extensive experimental studies on a Sun Blade/1000 workstation with a 750MHz UltraSPARC-III CPU running Solaris 2.8. The workstation has a total physical memory of 512M. We employed the  $TPCH$  (<http://www.tpc.org>) benchmark dataset, and conducted our testing using IBM DB2 version 7.1. The  $TPCH$  benchmark is a decision support benchmark for ad-hoc queries.

We mainly studied our hit prediction HP (Hit Prediction) in comparison with SFF (Small-Space-First) and SPF (Small-Penalty-First). We also compared our HP with the predicate-based approach [2]. In order to evaluate the performance

---

<sup>1</sup> In our testing, we set  $\gamma = 1$ .

of our approach, two performance metrics: hit ratio and cost saving ratio are used.

- **Hit Ratio**: it is the percentage of queries which can be answered by materialized views. A higher hit ratio represents more queries can be answered by the existing materialized views.
- **Cost Saving Ratio (CSR)** [4]: it measures the results as follows.

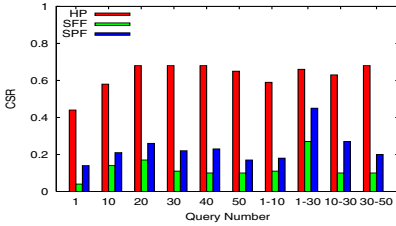
$$CSR = \sum_{i=0}^n \frac{wcost(q_i) - cost(q_i)}{wcost(q_i)} \times \frac{1}{n}$$

where  $cost(q_i)$  is the query processing cost using materialized view and  $wcost(q_i)$  is the query processing cost using base table in the data warehouse.

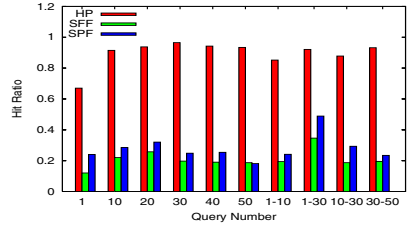
### 5.1 The Effect on Query Number per Batch

In this experiment, 600 queries are issued under 10% of base table cache space to investigate the effect on query number per batch. In Figure 1, 1, 10, 20, 30, 40 and 50 means each batch contains equal constant query number: 1, 10, 20, 30, 40 and 50 respectively. And 1-10, 1-30, 10-30 and 30-50 represent that each batch contains randomly generated query number which is between 1 and 10, 1 and 30, 10 and 30, and 30 and 50, respectively. Figure 1 shows that HP reaches a higher CSR and hit ratio than SFF and SPF no matter how many queries contained in a batch.

In the later experimental studies, we use 30 as a default query number for each batch.



(a) Effect of varying the query number per batch on CSR



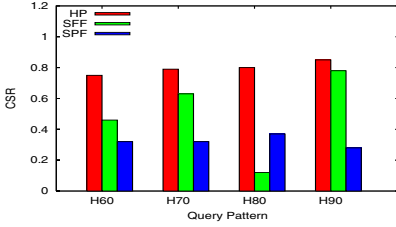
(b) Effect of varying the query number per batch on hit ratio

**Fig. 1.** Performance on varying the query number per batch.

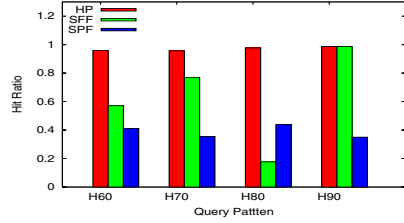
### 5.2 Query Locality

To study the query locality, we conduct two sets of experiments based on *data access locality* and *hierarchical access locality*. As most users have their own preferences which may last for a while, they may be interested in one part of data at one time. To simulate the data access locality, a certain percentage of

the database is designed as a hot region such that the queries are most likely to access the designated part of the database. Here, H60, H70, H80 and H90 represents 60%, 70%, 80% and 90% of the queries access 20% of the datacube, respectively. The rest of queries are uniformly distributed over the database. On the other hand, proximity queries are used to model hierarchical access locality. Q60, Q70, Q80 and Q90 mean 60%, 70%, 80% and 90% queries are proximity queries and 40%, 30%, 20% and 10% are random generated, respectively.

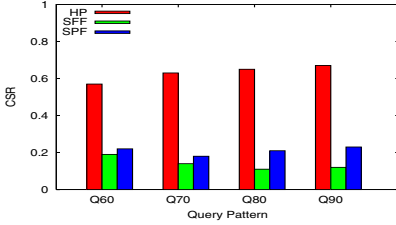


(a) Effect of varying the hot region on CSR

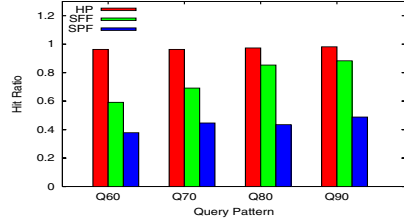


(b) Effect of varying the hot region on hit ratio

**Fig. 2.** Testing Different Data Access Locality Patterns.



(a) Effect of varying the proximity on CSR



(b) Effect of varying the proximity on hit ratio

**Fig. 3.** Testing Different Hierarchical Access Locality Patterns.

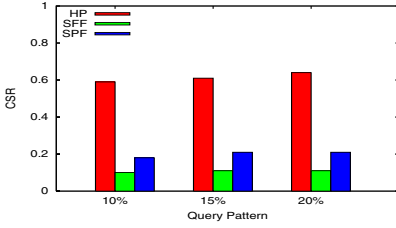
Assume that the cache space available is to hold 10% of the base tables. For each of the query patterns above, we issue 600 queries and calculate the CSR and hit ratio. Figure 2 and 3 show that the HP replacement policy exploits the locality very good. Figure 2 (a) shows the performance for query pattern with a designated hot region. CSR increases while more queries hit at the hot region of the database. Figure 2 (b) shows the hit comparison among HP, SFF, and SPF replacement policies. HP has significant hit ratio compared with other two approaches. Figure 3 shows the performance for proximity query pattern. In Figure 3 (a), CSR increases sharply as the proximity percentage increases. This is because more incoming queries can be derived from the cached query result. In Figure 3 (b), the HP replacement policy reaches a stable high hit ratio than other two replacement policies.

For simplicity, we choose *Q80* as the default query proximity pattern in the following experimental study.

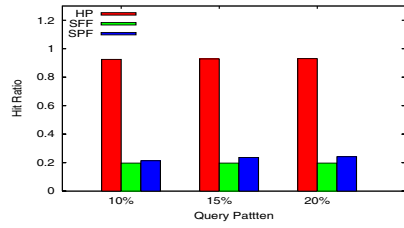
### 5.3 Query Sharing

In this experiment, given 10% of base table cache space, 600 queries are issued by varying the percentage of query sharing in 10%, 15% and 20%. That is, 10%, 15% and 20% of queries are the same.

As expected, Figure 4 (a) and (b) show that CSR and hit ratio increase while the sharing percentage increases. A higher sharing percentage means more queries can be answered by previous query result. The HP replacement policy outperforms than SFF and SPF.



(a) Effect of varying the share percentage on CSR



(b) Effect of varying the share percentage on hit ratio

**Fig. 4.** Performance on varying the query sharing percentage.

## 6 Conclusions

In this paper, we propose a new dynamic materialized view management with a new hit-prediction approach for caching OLAP queries. Our approach can detect and adapt the change of users' interest, and predict the probability of future reuse, such that it retains the most beneficial materialized views for fully utilizing the cache space during the dynamic materialized view management. The experimental results show that our replacement policy exhibits higher query locality and outperforms effectively and efficiently.

## Acknowledgment

The work described in this paper was supported by a grant from the Research Grants Council of the Hong Kong Special Administrative Region (CUHK4198/00E, HKUST6184/02E).



## References

1. E. Baralis, S. Paraboschi, and E. Teniente. Materialized views selection in a multidimensional database. In *Proceedings of the 23rd International Conference on Very Large Data Bases*, pages 156–165, 1997.
2. C.-H. Choi, J. X. Yu, and H. Lu. Dynamic materialized view management base on predicates. In *Proceedings of the 5th Asia Pacific Web Conference (APWEB)*, pages 583–594, 2003.
3. S. Dar, M. J. Franklin, B. T. Jónsson, D. Srivastava, and M. Tan. Semantic data caching and replacement. In *Proceedings of the 22nd International Conference on Very Large Data Bases*, pages 330–341, 1996.
4. P. Deshpande, K. Ramasamy, A. Shukla, and J. F. Naughton. Caching multidimensional queries using chunks. In *Proceedings of the 24th ACM SIGMOD International Conference on Management of Data*, pages 259–270, 1998.
5. A. Gupta and I. S. Mumick. *Materialized Views: Techniques, Implementations, and Applications*. The MIT Press, 1999.
6. H. Gupta, V. Harinarayan, A. Rajaraman, and J. D. Ullman. Index selection for OLAP. In *Proceedings of the 13th International Conference on Data Engineering*, pages 208–219, 1997.
7. H. Gupta and I. S. Mumick. Selection of views to materialize under a maintenance cost constraint. In *Proceedings of the 7th International Conference on Database Theory*, pages 453–470, 1999.
8. V. Harinarayan, A. Rajaraman, and J. D. Ullman. Implementing data cubes efficiently. In *Proceedings of the 22nd ACM SIGMOD International Conference on Management of Data*, pages 205–216, 1996.
9. P. Kalnis, W. S. Ng, B. C. Ooi, D. Papadias, and K.-L. Tan. An adaptive peer-to-peer network for distributed caching of olap results. In *Proceedings of the 28th ACM SIGMOD International Conference on Management of Data*, pages 25–36, 2002.
10. R. Kimball. *The Data Warehouse Toolkit*. John Wiley & Sons, 1996.
11. Y. Kotidis and N. Roussopoulos. Dynamat: A dynamic view management system for data warehouses. In *Proceedings of the 25th ACM SIGMOD International Conference on Management of Data*, pages 371–382, 1999.
12. W. Liang, H. Wang, and M. E. Orlowska. Materialized view selection under the maintenance time constraint. *Data and Knowledge Engineering*, 37(2), May 2001.
13. P. Scheuermann, J. Shim, and R. Vingralek. Watchman : A data warehouse intelligent cache manager. In *Proceedings of the 22nd International Conference on Very Large Data Bases*, pages 51–62, 1996.
14. A. Shukla, P. Deshpande, and J. F. Naughton. Materialized view selection for multidimensional datasets. In *Proceedings of the 24th International Conference on Very Large Data Bases*, pages 488–499, 1998.
15. D. Theodoratos and T. Sellis. Data warehouse configuration. In *Processings of the 23rd International Conference on Very Large Data Bases*, pages 318–329, 1997.

# A Cache Replacement Algorithm in Hierarchical Storage of Continuous Media Object\*

Yaoqiang Xu, Chunxiao Xing, and Lizhu Zhou

Department of Computer Science and Technology, Tsinghua University,  
Beijing, 100084, China  
Fax: +86-10-62771138  
xuyq99@mails.tsinghua.edu.cn  
{xingcx,dcsz1z}@mail.tsinghua.edu.cn

**Abstract.** Although storage technologies are in great progress, hierarchical storage is still an efficient way to manage massive data, especially massive Continuous Media (CM) data. In some large web based CM applications such as DL (Digital Library) and VOD (Video-On-Demand), cache replacement algorithms are used to keep the hot data on the disk, and migrate out those rarely accessed. The existing algorithms always treat all files equally, and use cache miss rate to evaluate the performance. However, in actual CM applications, most video objects are stored as several separated files that will be played one by one, so the strong relationships among the access time of these files make pre-fetch practicable and efficient. If all the files are taken as the same, the performance of systems will degrade. Meanwhile, the cache miss rate cannot evaluate the performance comprehensively. In this paper, a new metric, User Waiting Rate, is defined to evaluate the performance, and a novel cache replacement algorithm, Least Waiting Probability (LWP) algorithm, is proposed. Experiment results show that it can improve the performance a lot, and is highly adaptive.

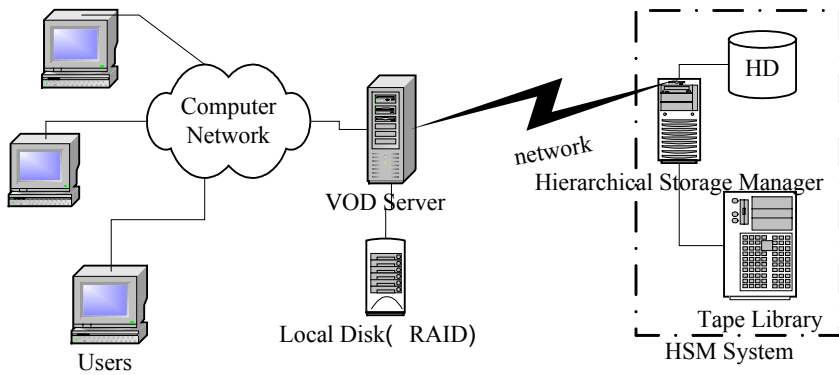
## 1 Introduction

Typical web based Continuous Media applications, such as Video-On-Demand, require very large storage capacity to store and manage tens of terabytes or even petabytes of data that can hardly be put on disk storage alone. A solution to this problem is to use Hierarchical Storage Management (HSM) Systems taking tertiary storage to expand capacity with reasonable performance assurance.

Fig. 1 shows the structure of a hierarchical storage based CM server. All the data resides on the HSM system (normally, in its tertiary storage devices). And the local secondary storage device (disk or RAID) of application server, which can be taken as the cache of HSM system, only holds the frequently accessed (hot) data. If the data requested by user exist in local disk cache, the request will be fulfilled immediately. Otherwise, data will be migrated from HSM system to disk cache. If there is no sufficient space in disk cache, cache replacement will be taken.

---

\* Supported by the National Natural Science Foundation of China under Grant No. 60221120146; the National Grand Fundamental Research 973 Program of China under Grant No.G1999032704.



**Fig. 1** Structure of Hierarchical Storage Based Media Application

Cache replacement algorithm is the key part in data migration. An ideal replacement algorithm is supposed to keep the data that WILL be accessed in cache, avoiding the long waiting of cache missing. Almost all the current algorithms such as LRU, LFU, ELFU, OLRU, etc. take the data as the same, and use cache miss rate as the only performance evaluation[1,5,7]. However, in actual CM applications, most video objects are stored as several separated files, which will be played one by one. There exist strong relationships among the access time of these files. And these make pre-fetch practicable and efficient. When users request an object, the playback can start immediately if the first file is in disk. For, the second file can be read during the play of the first file even if it is not in disk. Thus, taking all the files as the same will degrade the performance. At the same time cache miss rate cannot evaluate the performance comprehensively.

In this paper, User Waiting Rate is defined as the performance evaluation. By taking into account the relationship between the Expected Access Time of files and the access characteristic of HSM system, new file ranking function and data replacement algorithm are proposed.

The rest of this paper is organized as follows: section 2 introduces relative work, section 3 and 4 define User Waiting Rate and File Waiting probability. The new algorithm, LWP, is described in section 6. Simulation results and performance analysis is also given in this section. Section 7 is the conclusion.

## 2 Related Work

Cache and its data replacement algorithm are used on almost every level of the memory hierarchies. The most popular algorithms are LRU (Least Recently Used), LFU (Least Frequently Used), etc. Reed and Long introduced these algorithms in [1], and pointed out that LRU is a good algorithm in NFS server. LRU records the last access time for every data (block, or file), and migrates out the data with the oldest access time, while LFU records the access count for every data, and migrates out the data with least access count. LFU algorithm has a problem of cache pollution [2], that is, if an object once has been accessed many times and is not currently used, it may still

stay in the cache. A round robin like method was proposed in [3], it only records the access count in the recent period of time.

And there have been many researches on the replacement algorithms of disk cache in HSM systems. Lau and Liu proposed LEAT algorithm in [4], Yang, Ren and Chang described ELFU algorithm in [5]. Both of the two algorithms are based on block caching, and reduced the cache miss rate. For file-based caching, there are VBB algorithm [6], OLRU algorithm [7], etc.

File based caching is more popular in CM applications, for it's easier to combine current applications with HSM system. Besides, in CM applications, a CM object, say a movie, is always stored as two or more separated files. These files will be requested one by one, and so there is strong relationship between these files. It's a great pity that nearly all the algorithms do not utilize the relationship.

Other researches on hierarchical storage include data placement, scheduling, etc. Optimal data placement methods in tertiary storage libraries are presented in [8]. Li and Prabhakar study data placement using the information of objects relationship in [9]. We proposed a data placement method to reduce the system response time to almost 0 in [10].

### 3 User Waiting Rate

To make full use of the relationship among the files of a CM object, and improve the performance of the system by effective pre-fetching, User Waiting Rate (UWR) is defined to be the new metric of performance evaluation.

Let  $O_i$  denote the  $i$ th object in the system;  $F$  denote a file in the system;  $D$  denote the set of files located in the disk;  $F_{i,j} \in O_i$  denote  $F_{i,j}$  is the  $j$ th file of  $O_i$ . A file has attributes  $s$ ,  $EAT$ , and  $Pa$ . For  $F_{i,j}(s, EAT, Pa)$ ,  $s$  is the size,  $EAT$  is its Expected Access Time and  $Pa$  is the access probability of  $O_i$  and  $F_{i,j}$ .

**Definition 1.** File Ready Time (FRT): When fetching a file from HSM system, FRT is the time period between the time the request was sent and the time the file was ready in local disk for accessing. If pipeline mechanism is used, a file is ready if there is enough data in disk cache to start the pipeline. Otherwise, a file is ready only if the whole file is existed in disk cache.

**Definition 2.** Expected Access Time (EAT): If  $F_{i,j} \in O_i$ , then  $EAT(F_{i,j})$  is the time period from object  $O_i$  being requested to file  $F_{i,j}$  being requested. It can be calculated as follow:

$$EAT(F_{i,j}) = \begin{cases} 0, & j = 0; \\ EAT(F_{i,j-1}) + L(F_{i,j-1}), & j > 0; \end{cases}$$

Where  $L(F)$  means the playback time of file  $F$ .

When  $O_i$  is requested by a user, pre-fetch all the files  $F_{i,j}$  satisfy  $\forall F_{i,j}(s, EAT_{i,j}, Pa), F_{i,j} \in O_i$  and  $F_{i,j} \notin D$ , immediately. Let  $W_{i,j}$  denote  $FRT(F_{i,j})$ , if  $\text{Max}(W_{i,j} - EAT_{i,j}) < 0$ , which means all the files can be retrieved before its play,  $O_i$  can

be played immediately. Otherwise, in order to ensure the consistency of playback, the system must wait  $\text{Max}(W_{i,j} - \text{EAT}_{i,j})$  before playing the first file.

**Definition 3.** User Waiting Rate (UWR): UWR is the Rate of the count that a CM object cannot be played immediately on user request to the total request count.

UWR is counted from the aspect of the user. It can reflect the system performance more exactly and clearly. While, if cache miss rate is counted at the time of file accessing rather than object accessing, UWR is really the same as cache miss rate.

User Waiting Rate is used to evaluate the performance of the system in this paper, for it need not clarify the inspection time, and is logically clearer.

## 4 File Waiting Probability

File Waiting Probability (FWP) is presented to decide whether a file should be kept in the disk cache.

**Definition 4.** File Waiting Probability (FWP): For  $F(s, \text{EAT}, P_a) \in O$ , its FWP is the probability of  $\text{FRT}(F) > \text{EAT}$  when object  $O$  is requested by user and  $F \notin D$ .

FWP means the probability that the user needs to wait when the object is not in disk at the time requested. The file with larger FWP should be more likely to be kept in the disk cache than files with smaller FWP. Otherwise, UWR of the system will increase.

Assume the response distribution of HSM system is known, and  $R(t)$  denote the probability of  $\text{FRT}(F) > t$  when fetching file  $F$  from HSM system. Then the FWP of  $F(s, \text{EAT}, P_a)$  can be calculated as following:

$$\text{FWP}(F) = P_a * R(\text{EAT}) \quad (1)$$

In fact, the response distribution of HSM system is very complex. The approximation of  $R(t)$  will be discussed later. However, it is obvious that  $R(t)$  will reduce when  $t$  increases. So even when the access probability of  $O_a$  is larger than  $O_b$ , the result may be  $\text{FWP}(a,1) < \text{FWP}(b,2)$ , where  $\text{FWP}(i,j)$  denotes the FWP of the  $j$ th file of object  $i$ . So  $F_{b,2}$  should remain in the disk, not  $F_{a,1}$ .

## 5 Least Waiting Probability (LWP) Algorithm

Least Waiting Probability algorithm (LWP) uses File Waiting Probability (FWP) as the file ranking function. The file with least FWP will be replaced.

In this section, we will discuss the implementation of the algorithm, including the approximation of  $R(t)$ , calculation of  $P_a$ , and a simplification of the algorithm.

### 5.1 Approximation of $R(t)$

The major storage device in HSM system is tape library. As discussed in [10], tape library is an M/G/K queuing system (or harder). So there is no analytical method to get the mean queuing time and other values, neither  $R(t)$ . The only way available is simulation.

We use simulation similar to that discussed in [10] to get the distribution of  $R(t)$ . Figure 2 shows  $R'(t)$  which is the distribution of File Ready Time under different workloads when pipeline mechanism is used. A point  $(t, p)$  on the curve means that under that workload, the ratio of files with  $FRT \leq t$  to all requests is  $p$ . Obviously,  $R'(t) = 1 - R(t)$ .

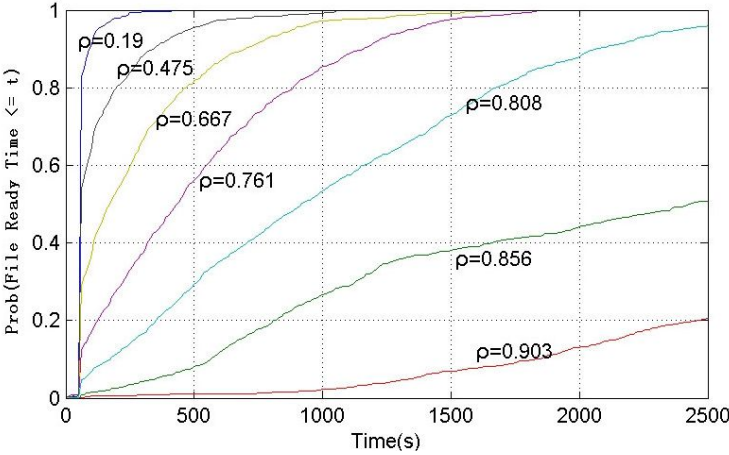


Fig. 2 Distribution of File Ready Time

We use lookup table to express these curves. Table 1 shows the lookup table corresponding to Fig. 2. The first column is the value of  $t$ , and the second column is the value of  $R'(t)$  when  $p=0.19$ , etc.

Table 1. Lookup table of  $R'(t)$

T	$p=0.19$	$p=0.475$	$p=0.666$	$p=0.761$	$p=0.808$	$p=0.856$
0	0	0	0	0	0	0
100	0.9115	0.6489	0.3713	0.1723	0.069	0.0149
200	0.9803	0.8053	0.5383	0.2809	0.1152	0.0255
400	0.9967	0.9245	0.7532	0.4702	0.2297	0.0606
500	1	0.9543	0.8181	0.5617	0.293	0.0809
800	1	0.9851	0.9319	0.7628	0.4468	0.1936
1000	1	0.9926	0.9713	0.8553	0.5329	0.267
1300	1	1	0.9862	0.9415	0.6468	0.3585
1500	1	1	0.9915	0.9766	0.7297	0.3798
2000	1	1	1	1	0.8797	0.4415
2500	1	1	1	1	0.9595	0.5085

For specified  $t$  and  $p$ ,  $R'(t, p)$  can be calculated as follows:

- 1) Find  $p_1$  and  $p_2$  which are the nearest two workloads to  $p$ ;
- 2) Find  $t_1$  and  $t_2$  which are the nearest two times to  $t$ ;
- 3) Get  $R'(t, p_1)$  from  $R'(t_1, p_1)$  and  $R'(t_2, p_1)$  using interpolation;
- 4) Get  $R'(t, p_2)$  from  $R'(t_1, p_2)$  and  $R'(t_2, p_2)$  using interpolation;
- 5) Get  $R'(t, p)$  from  $R'(t, p_1)$  and  $R'(t, p_2)$  using interpolation;

Then,  $R(t, p) = 1 - R'(t, p)$ .

The workload of HSM system is determined by not only the arrival rate of user request, but also cache replacement algorithm. When the user requests' arrival rate is approximately steady, the workload will be light if hot files are cached in disk. Otherwise, it will be heavier.

To get a dynamic balance, the workload  $\rho$  should be proportional to  $P_a$  when calculating the  $R(t, \rho)$  of  $F(s, t, P_a)$ .

Let  $\rho_0$  be the mean workload of HSM system,  $P_{a_0}$  be the mean access probability of files not cached in disks, then the corresponding workload of file  $F(s, t, P_a)$  can be calculated as:

$$\rho = \frac{\rho_0}{P_{a_0}} * P_a \quad (2)$$

## 5.2 LWP Algorithm

In LWP algorithm, we record the access count of file using the method similar to improved LFU. During the calculation,  $P_a$  is replaced by  $Ca$ , which is the access count and will decrease by 1 everyday.

The algorithm can be described as follows:

- (1) User request arrives. Update the access count record of the requested object  $O_i$ :  $Ca_i++$ . For every file belongs to  $O_i$ , update its File Waiting Probability.
- (2) If all files of  $O_i$  exist in the disk, apply the service immediately.
- (3) Otherwise, for every  $F_{i,j}(s, EAT_{i,j}, Ca)$ ,  $F_{i,j} \in O_i, F_{i,j} \notin D$ , begin to fetch it from HSM system, and get  $W_{i,j}$ , which is its File Ready Time. If all  $W_{i,j} < EAT_{i,j}$ , apply the service immediately; otherwise increase the waiting count, wait for  $\text{Max}(W_{i,j} - EAT_{i,j})$ , and then apply the service.
- (4) Every 24 hours, decrease the access record of every object by 1, and update the File Waiting Probability of every file.

$W_{i,j}$  is calculated by HSM system, which is possible when FIFO scheduling is applied.

## 5.3 The Simplified LWP (SLWP) Algorithm

The core idea of LWP is to migrate out the latter files of some hotter objects, and cache the first files of some colder objects. But it's difficult to provide the distribution of FRT. And the distribution needs to change with the configuration of the HSM system. To migrate out the latter files, algorithm can be simplified as follows if every object contains two files with the same size: Suppose the disk can cache  $N$  objects totally. We sort objects by their access count in descending order, and define  $\Delta$  be the exchange factor ( $0 \leq \Delta < 1$ ). Then:

- For the hottest  $N(1-\Delta)$  objects, both of the files are cached in the disk;
- For objects after  $N(1+\Delta)$ , both of the files are stored in HSM system;
- For Objects between  $N(1-\Delta)$  and  $N(1+\Delta)$ , the first file is cached in the disk.

6 Simulation Result

To compare the performance of different algorithms, simulation program is designed using Sim++[12,13] for a VOD system. Figure 3 shows the queuing model of the system.

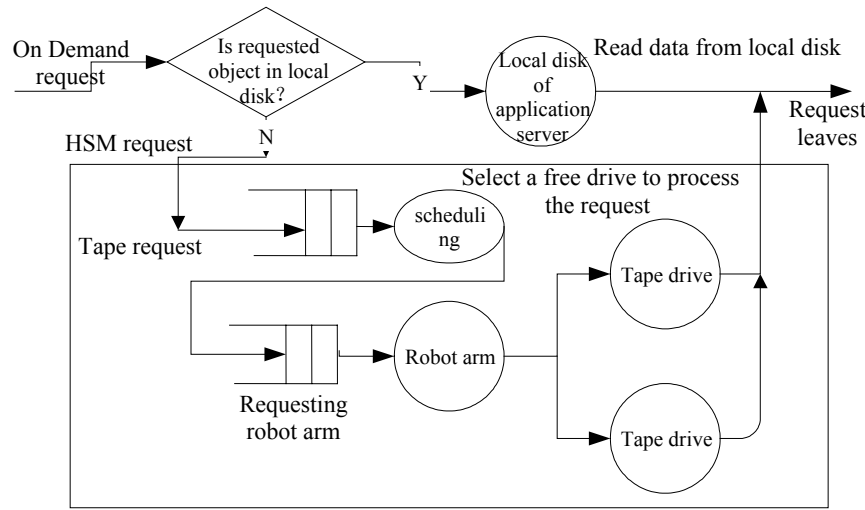


Fig. 3 Queue Model of The System

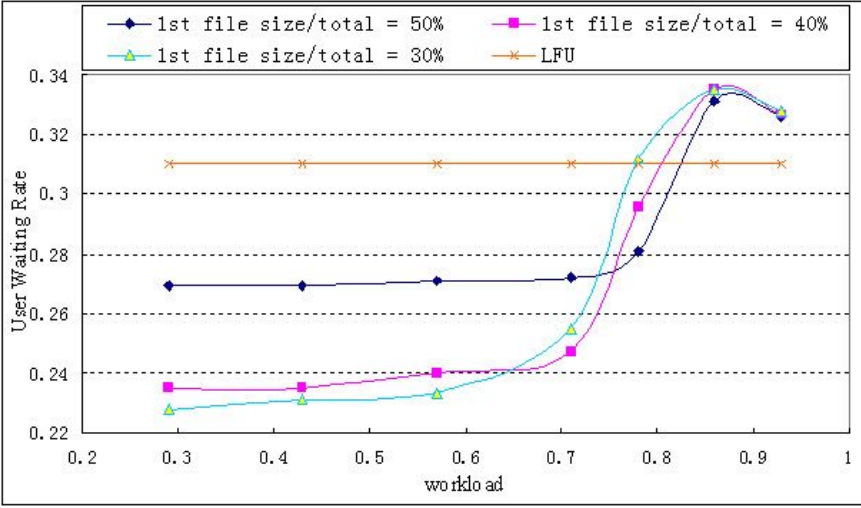
The arriving time of users' requests has exponential distribution, and the object requested has Zipf distribution. The tape library in HSM system is Sony DTF and the robot arm is Sony DMS-B9. The main parameters are listed in the table 2.

Table 2. Main Parameters Of The System

Symbol	Value	Note
TapeLoad_time	12.8s	Time for robot arm to load a tape
TapeUnload_time	17.2s	Time for robot arm to unload a tape
TapeMount_time	51s	Time period between the time when the tape is placed into the drive and the time when the tape can be operated
TapeUnmount_time	18s	Time period between the time when the "unmount" command is issued and the time when the tape is extracted from the drive by the robot arm
Transfer_rate	12MBps	Persistent read/write rate of the tape
Seek_rate	300MBps	Quick seek rate of the tape
Tape_size	42G	Tape capacity
Object_size	2G	Size of each media object.
Tape_count	60	Total tape count in the tape library
Drive_count	2	Drive count in the tape library
Object_count	2000	Total media object count in the system
CachedObject	100	Object count that can wholly stored in the disk



In the experiment, each media object contains two files. Their size may be different in LWP algorithm, but be equal in SLWP algorithm. We compare the LFU, LWP and SLWP algorithms in various workloads.



**Fig. 4** Performance Result of LWP Algorithm

Fig. 4 shows the User Waiting Rate of LWP algorithm. When the workload is not very heavy ( $\rho < 0.8$ ), LWP has good performance. If the two files of the object have the same size, LWP can reduce the UWR by 13%. And if the first file is smaller than the second file, the reduction may be greater than 26%.

While, if the workload turns to be heavier, the UWR of LWP increases quickly, and may be worse than LFU. The reason is that the performance of HSM system changes with workload very rapidly when the workload is heavy. LWP algorithm replaces hot files from disk cache out to HSM system, thus increase the workload of HSM system.

Fig. 5 shows the performance of Simplified LWP algorithms with different  $\Delta$ , comparing with LWP and LFU algorithms. When the workload is low, SLWP performs well and may reduce the UWR by 18.3%. A larger  $\Delta$  may cause the better performance in low workload. However, in heavy workload, for the same reason as LWP, the performance of SLWP turns to be worse quickly.

Fig. 5 also compares the performances of SLWPs with LWP. We can see that the performance of LWP is better in most cases. And its effective range is larger than those of SLWPs. By changing the  $\Delta$ , we find that the 0.23 is the best balance point for SLWP in that saturation. The curve of LWP is almost the same as SLWP with  $\Delta=0.23$  in low workload, and it performs better than SLWP with  $\Delta=0.1$  in heavy workload. So we can say that LWP algorithm has strong self-adaptive ability. It finds good exchange factor for specified workload automatically ( $\Delta=0.23$  for low workload, and  $\Delta < 0.1$  for heavy workload). Because of the complexity of HSM system, the adaptive ability of LWP algorithm is not very ideally in heavy workload. But in actual CM applications, the workload will never be too heavy to ensure a reasonable performance.

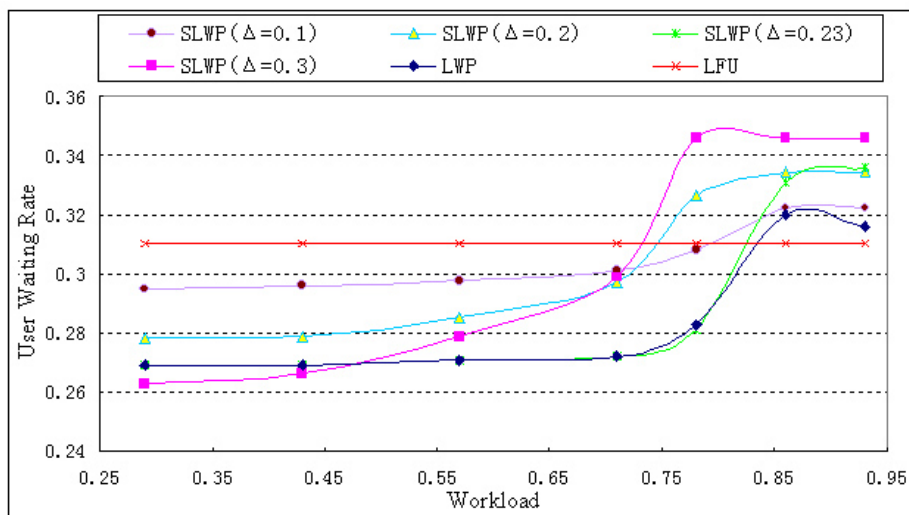


Fig. 5 Performance Result of SLWP Algorithm

This result also points out that File Waiting Probability is a good file ranking function.

## 7 Conclusions and Future Work

In the hierarchical storage system of CM files, by analyzing the characteristics of CM applications, the User Waiting Rate is proposed to evaluate the performance of different cache replacement algorithms and a novel cache replacement algorithm, Least Waiting Probability algorithm, is designed. The User Waiting Rate of LWP is lower for about 13% than LFU. The Simplified LWP performs well too.

A better response distribution curve of HSM system may be found in future work to improve the performance of LWP. Besides, a self-adaptive mechanism of the factor in the Simplified LWP may be designed to make it perform well under various workloads.

## References

1. B. Reed, D.D. Long, Analysis of caching Algorithms for distributed file systems, Operating Systems Review, 30(3), 1996
2. R. Karedla, J. Love, and B. Wherry. Caching Strategies to Improve Disk System Performance. IEEE Computer, 1995, 27(3):38--46
3. LI Yong, CHEN Fu-jie, WU Fei. Design and Management of Large Scale Hierarchical VOD Storage System. Journal Of Software (in Chinese), 1999, 10(4):355-358
4. Siu-Wah Lau, John C. S. Liu. Designing a hierarchical Multimedia storage server. The Computer Journal, 1997, 40(9):529-540

5. Yang Daoliang, Ren Xiaoxia and Chang Ming. Study on Data Replacement Algorithm in Continuous Media Server with Hierarchical Storage. 16th IFIP World Computer Congress, Beijing, 2: 1387-1394, 2000
6. Sonah, B., Ito, M.B. New adaptive object replacement policy for video-on-demand systems. In Proc of IEEE International Workshop on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems, 13-18, 1998
7. Ulrich Hahn, Werner Dilling, Dietmar Kaletta. Improved adaptive replacement algorithm for disk caches in HSM systems. In the 16th IEEE Symposium on Mass Storage Systems, 1999
8. S. Christodoulakis, P. Triantafillou, and F. Zioga, "Principles of Optimally Placing Data in Tertiary Storage Libraries", In Proc. of the 23rd Intern. Conf. on Very Large Data Bases, (VLDB) August 1997
9. Jiangtao Li, Sunil Prabhakar. Data Placement for Tertiary Storage. In the 19th IEEE Symposium on Mass Storage Systems, 193-207, 2002
10. Yaoqiang Xu, Chunxiao Xing, Lizhu Zhou. A Data Placement Method of HSM for Streaming Media Server in Network Environment. In Proc. of 3rd Workshop on Databases in Networked Information Systems, 245-254, 2003
11. Shahram Ghandeharizadeh, Ali Dashti, and Cyrus Shahabi. A pipelining mechanism to minimize the latency time in hierarchical multimedia storage managers. Computer communications, 18(3): 170-184, 1995
12. Paul A. Fishwick. Simpack: Getting Started with Simulation Programming in C and C++. In 1992 Winter Simulation Conference, December 1992, Arlington, VA, 154 - 162
13. SimPack Toolkit. <http://www.cise.ufl.edu/~fishwick/simpack/simpack.html>

# CCAN: Cache-Based CAN Using the Small World Model\*

Futai Zou, Yin Li, Liang Zhang, and Fanyuan Ma

Department of Computer Science and Engineering  
Shanghai Jiao Tong University, 200030 Shanghai, China  
zoufutai@sjtu.edu.cn

**Abstract.** The small world model has been used to improve the routing performance in P2P networks recently. In this paper, we proposed CCAN, a cache-based CAN, which is based on the small world model. CCAN caches the long contact and takes a novel probabilistic cache replacing strategy. The probabilistic cache scheme proves to be a new approach to model the small world phenomenon. Experiments in both the static and the dynamic network show that CCAN can be converged to a stable status with the cache scheme, and the routing performance is improved with almost zero overheads in the network compared with CAN.

## 1 Introduction

Peer-to-Peer (P2P) overlay networks have been growing rapidly in the last few years. It aims to aggregate large numbers of computers that join and leave the network frequently and represents the notion of sharing resources available at the edges of the Internet.

The existing peer-to-peer networks can be broadly classified as either unstructured or structured. Resource location in unstructured P2P networks is generally based on flooding [1] or random-walkers [2]. It arises too many query messages and is not deterministic to find the resource in the network. In contrast to this, structured P2P networks can provide a deterministic resource location using DHT (Distributed Hash Table) schemes but have to face the high overhead of maintaining DHT routing data structures. This paper focuses on structured P2P Networks and investigates how to realize an efficient resource location with low maintaining overheads on CAN [3].

Our ideas are originated from the small world phenomenon in P2P networks [4] and illuminated by Kleinsberg's recent work [5]. Kleinsberg generalizes a class of small network model. This model proposes that node  $S$  should choose long range contact  $T$  with the probability of  $L^{-r}$ , where  $L=||S-T||^1$  is the *Manhattan distance* between node  $S$  and  $T$ , and  $r$  is the dimension degree of the underlying topology. The

---

\* Research described in this paper is supported by The Science & Technology Committee of Shanghai Municipality Key Technologies R&D Project Grant 03dz15027 and by The Science & Technology Committee of Shanghai Municipality Key Project Grant 025115032.

<sup>1</sup> We define  $||n1-n2||$  as the Manhattan distance between node  $n1$  and node  $n2$ . In a  $d$ -dimensional grid with two point  $X$  and  $Y$ , the Manhattan distance is  $\sum_{i=1}^d |X_i - Y_i|$ .

distribution of long contact is called the *inverse  $r^{\text{th}}$ -power distribution* by Kleinsberg. It gives an upper-bound path length  $O(\log^2 n)$  even with one long range contact.

Kleinsberg's construction is especially useful for constructing and improving the low-degree P2P systems. There are a serial of Kleinsberg-style networks such as [6] [7]. It is an intuitive idea to construct the small world on CAN because CAN has a topology similar with Kleinberg's model. However, Kleinsberg's model is a static model which uses the global information of all nodes to construct the small world phenomenon. In the contrast, nodes join and leave freely in P2P systems and each node has just a little information about the whole system. For example, each node only knows its  $2d$  neighbor nodes and nodes join and leave freely in CAN. Therefore we try to use the technique of cache nodes with a probabilistic cache replacing strategy to model the small world phenomenon instead of the static construction in Kleinsberg's model. The potential benefit is an improved performance with almost zero overheads in the network. Because the cache technique is critical for our design, so we name it CCAN (Cache-based CAN).

The remainder of the paper is organized as follows. Section 2 presents CCAN system model. The feasibility of cache mechanism and routing performance is simulated in Section 3. We conclude our research in the last section.

## 2 CCAN

In this section, we discuss the topology and the cache mechanism of CCAN.

### 2.1 Topology

CCAN equips peer nodes on a  $d$ -torus,  $d$  is the positive integer and presents the dimension degree of the topology. CCAN uses DHT scheme and the entire CCAN space is divided among the nodes currently in the system. Each object is assigned a key uniformly chosen from the key space  $M$ . Assume  $n$  nodes in the node space  $N$  and  $N \subset M$ . Each node stores objects that correspond to a certain portion of the key space and uses a routing table to forward the request for an object not belonging to its key space to appropriate "next hop" nodes.

The routing table of CCAN includes two parts: *local neighbors* and *cached long contacts*. First we just discuss one cached long contact, and then we would extend it to multi cached long contacts. CCAN uses the greed routing algorithms and a query message is forwarded to the node that is the closest to the query id.

### 2.2 Cache Replacing Scheme

When the system runs, a node  $S$  answers a query request from node  $T$ . Suppose node  $S$  has already node  $K$  as its cached long contact in the cache. Upon answering the request from  $T$ , it replaces  $K$  with  $T$  with probability  $\|S-K\|^d / (\|S-K\|^d + \|S-T\|^d)$ . Otherwise it keeps the key. During the routing process, the forwarding nodes always use the original node to do the cache replacement. For the approach is like a worm model, we call it the *worm cache replacing mechanism*.

When one node receives a query request message, first, it uses its routing table to forward this message, and then uses the message to update its cache. It is almost zero overheads attached to the node. Furthermore, a worm cache replacing process can avoid the hot key and make the replacing process faster.

Intuitively, a small phenomenon may exist because the replacement is related with the position in the underlying topology and favors the short distance nodes, which is the spirit of Kleinsberg's model. Here we prove the validity of this intuition by the following theorem:

**Theorem 1.** *Repeating the cache replacing procedure in the system. Considering the node space  $N$ ,  $\forall s \in N$ ,  $s$  would cache  $t$  ( $t \in N \cap t \neq s$ ) with probability proportional to  $\|s-t\|^{-d}$  in CCAN with  $d$  dimension degree.*

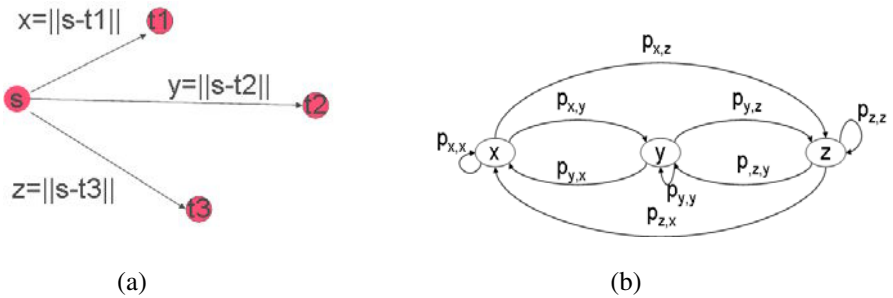
**Proof.**

Assume there are  $n$  nodes in the system. For  $\forall s \in N$  and  $t$  ( $t \in N \cap t \neq s$ ), we discuss the probability that  $s$  caches  $t$ .

We define a Markov chain as follows:

There are  $m$  kinds of distance from  $s$  to  $t$  ( $m$  is less or equal to  $n$  because there may exist some same distance away from node  $s$ ).

Let each kind of distance be a state. So the state space for node  $s$  is  $S = \{d_1, d_2, \dots, d_i, \dots, d_m\}$ ,  $d_i$  is the distance away from  $s$ . Suppose node  $s$  caches node  $t$  with the distance  $x = \|s-t\|$  and then we say that the Markov chain is in state  $x$ . A step is made when a new sample (on the arrival of a query routing message) with distance  $y$  is received. Then the chain walks to state  $y$  with probability  $x^d / (x^d + y^d)$ , otherwise it stays in the same state. This procedure clearly defines a regular Markov chain with  $m$  states. Fig. 1 shows an example of this Markov chain with only 3 states for node  $s$ , which means there are some nodes with three kinds of different distance away from node  $s$ . For a simplified illustration, the example in Fig.1 shows only 3 nodes  $t_1, t_2, t_3$  with different distance away from node  $s$ .



**Fig. 1.** A simplified Markov chain model with only 3 states for node  $s$ . (a) The node topology for node  $s$  with three neighbor nodes. (b) The state transition diagram.

So a stable and unique probability distribution for the process exists for any finite number of states  $m$ .

Suppose the stable probability for an arbitrary state  $x$  be  $p_x$ . Let us consider the flow-in and flow-out process for the state  $x$  under the stable condition. It is important

that  $p_x$  is a stable value, therefore there is a state balance between flow-in and flow-out. This balance means that the total flow-out probability from state  $x$  to state  $i$  ( $i \in S, i \neq x$ ) is equal to the total flow-in probability from state  $j$  ( $j \in S, j \neq x$ ) to state  $x$ . Fig. 2 shows the state balance between flow-in and flow-out.

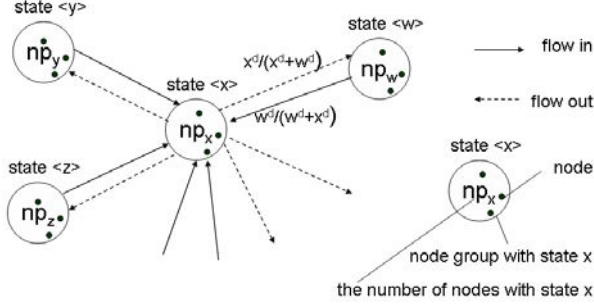


Fig. 2. The state balance between flow-in and flow-out.

According to Fig. 2, we have the equation •the left of the equation is flow-out and the right of the equation is flow-in)

$$np_x \sum_{i \in S} \frac{x^d}{x^d + i^d} = \sum_{j \in S} np_j \frac{j^d}{j^d + x^d} \quad (1)$$

Hence, evidently  $p_i = (1/c) \times \frac{1}{i^d}$ ,  $i \in S$  satisfies the equation where  $c = \sum \frac{1}{i^d}$  is

a normalizing constant. It indicates  $p_i$  is inverse proportion to the distance and the distribution follows up the inverse  $r^{\text{th}}$ -power distribution proposed in Kleinsberg's model. For all state  $x$  ( $x \in S$ ), it has the stable probability  $p_x$ . Notice that a state  $x$  means a distance between node  $s$  and node  $t$  and  $x = \|s - t\|$ , therefore, for  $\forall s \in N$ ,  $s$  would cache  $t$  ( $t \in N \cap t \neq s$ ) with probability proportional to  $\|s - t\|^{-d}$  along the repeating cache replacing process.

Since the stationary state is unique the system always converges to the proper distribution. Meaning that no matter what the initial cached long contact is and the system eventually forms towards the desired small world network as soon as enough random query requests are made.

### 2.3 Dynamic Convergence

Though we have proved that there has the small world phenomenon with the repeating cache replacing process in Section 2.2, it just considers the static situation. Here we discuss the dynamic situation where the nodes join and leave dynamically.

For the proof of Theorem 1, it assumes that there are  $n$  nodes in the system. Now the assumption is extend to  $N$  (the whole node space) nodes in the system. The same result in Theorem 1 can be drawn out following the similar process. Similarly, ran-

domly select  $n$  nodes from the node space  $N$ , the same result can be gotten. In fact, if there are enough cache replacing times, node  $s$  would cache an arbitrary selected node  $t$  from the node space  $N$  with the probability proportion to the distance  $\|s-t\|^{-d}$ . Therefore the system would converge to the small world network but with a fluctuating model due to the dynamic environment.

The difference from the static situation is that the convergence would be slow because nodes are dynamically changed. To accelerate the converging process, we install an *active query procedure* for each node. The procedure is invoked in an interval time which may be adjusted according to the dynamic of the network.

## 2.4 The Speed of Convergence

We give the qualitative analysis about the speed of convergence. The convergence is driven by the routing query message. The more messages the node forwards, the faster the convergence is.

Because the query is the uppermost operation in P2P networks and the cache uses the worm routing replacing mechanism we can expect that the system will be converged soon. The behavior of “hot key” query doesn’t mean an inconsistent cache process. Though the key may be very popular who receives many requests, the nodes initiating the request for that key can be assumed to be uniformly chosen from the network. With the worm routing replacing mechanism, the cache replacing possibility for each node is approximately even. Furthermore, we use the active query procedure to accelerate the convergence. It is especially useful when the network is dynamic.

## 2.5 The Maintenance of Routing Table

The maintenance of local contacts is same with CAN. The reader can refer to the [3] for more details. The cached long contact has two statuses: *active* and *invalid*. It would be checked whether it is still active within a scheduled period. Also the routing query message has the effect to change the cache from invalid to active with a corresponding probability. Thus to shorten the invoked interval of the active query procedure will help the maintenance of routing table.

**Theorem 2.** *With the worm routing cache replacing mechanism, CCAN system presents the small world phenomenon with the expected number of hops is  $O(\log^2 n)$ . Here  $n$  is the number of nodes in the system, and  $d$  is the dimension degree of CCAN.*

### Proof.

The construction of CCAN is similar with Kleinsberg’s small world construction. Both have the similar local contacts and a long contact. Though the long contact of CCAN is in cache, it has proved by Theorem 1 that both cached long contact would follow up the same  $r^{\text{th}}$ -power distribution with the worm routing cache replacing mechanism in a feasible approach. Thus the expected number of hops is  $O(\log^2 n)$  following from a similar theorem in [5].



## 2.6 Multi-cached Long Contacts

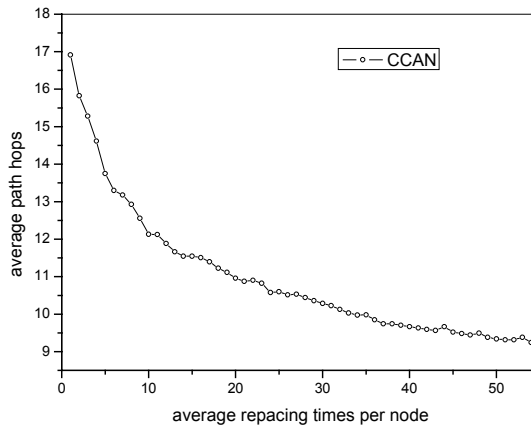
Above we just consider the situation of one cached long contacts. Now we extend to the situation of the  $k$  ( $k \geq 1$ ) cached long contacts. In this time, when a routing message reaches, it replaces all the cache with the same original node in the query message so as to keep convergence speed similar with the case of one cached long contact. The  $k$  cached long contact can be treated as  $k$  states in the Markov chain described in the proof of Theorem 1 and thus it is still a small world construction for the system. With the incremental cached long contacts, it has more choice for the routing for a query thus average path hops would be reduced.

## 3 Experiments

In this section, we present results from our simulation of CCAN. We tested for the convergent process in static and dynamic network respectively and the improved performance compared with other approaches. The node takes its introducer as its initial cached long contact and updates its cache with our worm cache replacement mechanism while forwarding the message once a query message arrives. Simulations are based on a 2-dimensional CCAN where the topology is much similar with Kleinsberg's construction. CCAN runs on the network ranging from 64 to 4096 nodes in size with the random distribution of these nodes in the network.

### 3.1 Static Convergence

We consider the ideal situation that there isn't any node failure in the network. After 1024 nodes are joined into the network, two nodes are randomly chosen from the network and one initiates a query for the other. The routing hop number and the times of cache replacement are recorded in every query. After 100 queries are made, the average path hops and the average replacing times per node are counted. Fig. 3 plots the gradually converging process. As clearly seen in the figure, the system totally

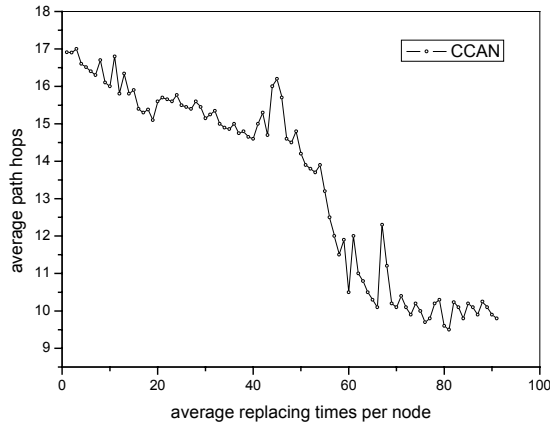


**Fig. 3.** The static converging process for CCAN. Each data point is the average of 100 queries.

reaches the steady when the average replacing times are 50 times of the total number of nodes. This is a totally satisfying result, meaning that each node needs 50 steps for the stabilization of its cache replacing Markov chain so that the small world network is formed.

### 3.2 Dynamic Convergence

For a reality network, nodes join and leave dynamically. We simulate this phenomenon as follows. After 1024 nodes are joined into the network, some nodes leave the network and some nodes join the network and the join and leave with the same rate. In this way, the total number of nodes is still invariable which is a generally assumption for the realistic network. To form the small world, the cache replacing mechanism should let an arbitrary new joining node has the possibility to be stable. The rate for join and leave is one per ten minutes. We employ the active query procedure proposed in Section 2.4. The active query procedure can accelerate the convergence and help to modify some invalid caches with a proper probability. From Section 3.1 it needs about 50 steps to make Markov chain stable. Therefore we set the active query procedure with an interval time of 10 second so that it can be sent 60 times query from each node by the procedure in the live time of the node, and in this way it is possible to make a better convergence for the dynamic network. The interval time of the active query procedure installed in nodes should be adjusted according to the dynamic degree of the network. In this paper, it is configured as a system argument. Also it could be a self-adjusting argument with a few modifications.



**Fig. 4.** The dynamic converging process for CCAN. Each data point is the average of 100 queries.

Similarly, two nodes are randomly chosen from the network and one initiates a query for the other. The routing hop number and the times of cache replacement are recorded in every query. After 100 queries are made, the average path hops and the average replacing times per node are counted. For the dynamic model, however, we only consider the successful query for the count of the average path hops and current

alive nodes for the count of the average replacing times. The result is depicted in Fig. 4. Obviously, it is more dynamic and slow to be converged into the stable status compared with the static model. The result shows that the system always moves towards the desired construction. The reason is that the probabilistic cache replacing mechanism always favors to the short distance contacts so that the distribution of long contacts can follow up the inverse  $r^{\text{th}}$ -power distribution. Therefore the topology is gradually adjusted towards the small world network.

### 3.3 Path Hops

CCAN has the improved performance with the cached long contacts. The experiments have presented that CCAN can be converged to the desired small world topology. Thus, path hops should be reduced with this topology. The simulation runs in the networks ranging from 64 to 4096 nodes in size and the average path hops for the queries are tested under the stable system. The path hops is evaluated with different approaches. We varied the probability for cache replacement and the number of cached long contacts. The results are showed in Fig. 5. First of all, path hops has been reduced with CCAN compared with CAN (it is also a 2-dimensional topology). This is guaranteed by Theorem 2. At the same time, we observe that the incremental number of cached long contacts help to improve the routing performance because more routing choices are provided with them. At last, as proved in [5], a random method doesn't take the distance or the position into consideration so that it can not form a small world phenomenon. Therefore the path hops cannot be improved in CCAN with random replacement showed in Fig. 5.

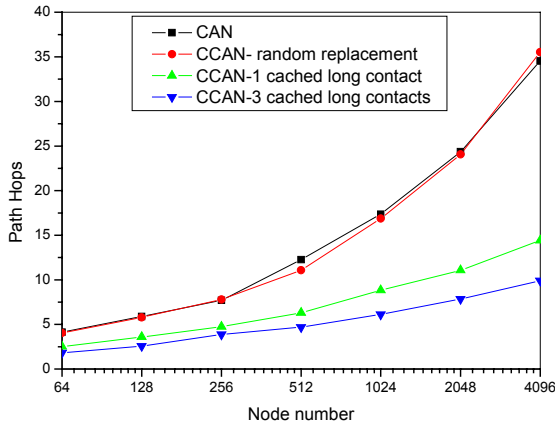


Fig. 5. Comparing path hops with different approaches.

## 4 Conclusions

In this paper, we propose CCAN to improve CAN performance using small world model. In CCAN, a novel probabilistic cache scheme on the long contact is used to

model the small world phenomenon. With the probabilistic cache replacing process, we prove that the cached long contact would follow up the same inverse  $r^{\text{th}}$ -power distribution in Kleinsberg's model. Then we discuss the feasibility of this approach in dynamic P2P networks and emphasize on how to accelerate the convergent process. Our experiments show the convergence in both static and dynamic network. Therefore the system would always converge towards the stable status with the desired small world network. Average path hops in different configurations are simulated under the stable status. The results show that path hops are reduced as we have expected.

CCAN can improve path hops from  $O(dn^{\frac{1}{d}})$  to  $O(\log^2 n)$ . For a low dimensional CAN, the improvement is significant.

The probabilistic cache scheme is successfully introduced into CAN in this paper. However, it is easily to be extended to a class of P2P systems with low-degree routing networks. The routing table has a few local contacts in these systems and can form a small world search network with additional cached long contacts according to our scheme. These systems don't need to be modified too much because the cache is managed locally. The main limitation in our cache scheme lies in that it depends on the relatively dense and frequent query. Therefore our scheme is especially feasible and preferable for the systems with huge query messages such as resource sharing P2P systems, which are very popular in today's applications.

## References

1. Gnutella. <http://www.gnutella.co.uk>
2. Qin Lv, Scott Shenker et al. Search and Replication in Unstructured Peer-to-Peer Networks. In Proceedings of ACM SIGGRAPH', 02 Aug. 2002.
3. RATNASAMY, S., FRANCIS, P., HANDLEY, M., KARP, R., AND SHENKER, S. A scalable content-addressable network. In Proc. ACM SIGCOMM (San Diego, CA, August 2001), pp. 161–172.
4. A. Iamnitchi, M. Ripeanu, and I. Foster, Locating Data in (Small-World?) Peer-to-Peer Scientific Collaborations, 1st International Peer To Peer Systems Workshop (IPTPS 2002).
5. J. Kleinberg. The small-world phenomenon: an algorithmic perspective. Cornell Computer Science Technical Report 99-1776, 2000.
6. G. S. Manku, M. Bawa, and P. Raghavan. Symphony: Distributed hashing in a small world. Proc. 4<sup>th</sup> USENIX Symposium on Internet Technologies and Systems (USITS 2003), 2003.
7. D. Malkhi, M. Naor, and D. Ratajczak. Viceroy: A scalable and dynamic emulation of the butterfly. In Proc 21st ACM Symposium on Principles of Distributed Computing (PODC 2002), pages 183–192, 2002.

# Performance Evaluations of Replacement Algorithms in Hierarchical Web Caching\*

Haohuan Fu, Pui-On Au, and Weijia Jia

Department of Computer Engineering and Information Technology  
City University of Hong Kong, Hong Kong SAR, China  
fu.haohuan@student.cityu.edu.hk  
{itpoau,itjia}@cityu.edu.hk

**Abstract.** Web caching plays an important role in many network services. Utilization of the cache in each level (server, proxy, and client) of network forms a web caching hierarchy. A major problem of the hierarchical caching is the poor performance due to the interference between upper-level and lower-level caches. This paper investigates the replacement algorithms applied in the network caching hierarchy through trace-driven simulation experiments to identify the performance bottleneck. Our investigation focuses on three fundamental replacement algorithms: LRU, LFU and SIZE, because many other replacement algorithms are mainly the variations and/or combinations of the three fundamental algorithms. Through extensive experiments, we have acquired useful performance features of these algorithms and their combinations at different levels. Thus, our work may serve as a reference for the design and deployment of efficient replacement algorithms in the caching hierarchy for web engineering.

## 1 Introduction

Cache technology, which was adopted into the web applications from memory systems, has been proved to be a very useful approach to solve the latency and scalability problems of web services. Nowadays, caching can be found in almost every level of the web accessing. At the bottom of the network hierarchy, there is client-side cache in the browsers (even for the first-generation web browsers, like Mosaic [1], there is cached data for the future references). Server-side caching is normally on the top level, which is usually much more comprehensive and faster than the client-side counterpart. In addition, there may be several levels of proxy caches in between of the server and clients. Therefore, the server-side cache, the proxy cache and the client-side cache, consist the network caching hierarchy.

In this paper, we discuss the performance of the replacement algorithms in the caching hierarchy, especially the performance on the higher levels. Our investigation focuses on three most basic algorithms: LRU, LFU and SIZE [14]. With extensive simulation experiments, we are trying to reveal some useful knowledge about the

---

\* The work is supported by Research Grant Council (RGC) Hong Kong, SAR China, under grant nos.: CityU 1055/00E and CityU 1039/02E and CityU Strategic grant nos. 7001446 and 7001587.

working pattern of these different algorithms in the caching hierarchy environment to facilitate the selection and adjustment of using the replacement algorithms.

The rest of the paper is organized as follows. Section 2 gives related work about hierarchical caching. Section 3 discusses some general problems we need to consider when applying the replacement algorithms in the caching hierarchy environment. Experiment design, results and analyses are given in section 4. Section 5 concludes the paper.

## 2 Related Work

There are many well-known systems about the hierarchical caching such as IRCache [2], which provided operational hierarchical caching services for organizations and individuals. There was an experimental project of national-wide hierarchical cache system in UK in 1995 [3], proposed by HENSA UNIX. And in recent years, there is the JANET Web Caching Service (JWCS), which is a national caching service and the hub in UK Higher Education caching infrastructure [4]. However, poor performance at the upper level has always been a problem for these kinds of systems. The performance is becoming poorer and poorer from the bottom to the top level caches.

### 2.1 Some Existing Work

Compared with the hierarchy structure, a distributed structure has been suggested to improve the performance. In [6], Metadata is suggested to track copies of files stored in a distributed cache system. And in the suggestion of [7], only the lower level caches in the hierarchy are responsible for caching data, and the upper level caches are used to store the information about the content of the lower level caches. Caching Array Routing Protocol (CARP [8]) provides the distributed structure for different proxy caches, which are configured as one logical cache for users.

Cooperation between caches is also suggested for different levels. In [9], a collaborative method is proposed for the hierarchical caching. The upper level caches pass the caching information to the lower level, so that the caches at the lower level can make some better caching decisions. The cooperation can also occur between caches at the same level. Fan et al. [10] suggested a cache sharing protocol, in which each proxy stores a summary of URLs of documents cached at every other proxy so that missed requests can be sent to other proxies which have copies of the requested files.

Using different replacement algorithms at different levels, as suggested in [5], will also improve the performance of caching hierarchy. However, [5] failed to provide a clear analysis of the different algorithms' working characteristics in the hierarchy environment.

### 2.2 Replacement Algorithms

For the replacement algorithms, three fundamental algorithms LRU, LFU and SIZE, have been widely used. LRU stands for Least-Recently-Used, which means that the least recently accessed objects will be evicted from the cache. It uses the last access

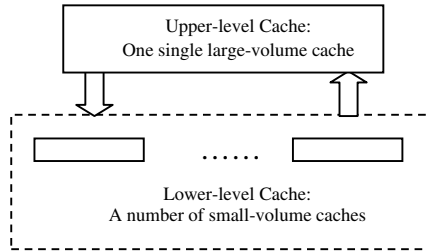
time (ATIME) of the data items as the sorting keys. LFU stands for Least-Frequently-Used, which means that the least frequently accessed objects will be evicted and the number of references (NREF) is used as the sorting key. SIZE [14] algorithm simply uses the size of the data items as the sorting key and the data items with the largest size is evicted from the cache.

There are still other different kinds of replacement algorithms. Some are direct extensions of the basic LRU, LFU algorithms, such as LRU-MIN [12], SLRU [11], LFU-Aging [13] and so on. Some are key-based policies, such as Hyper-G [14], which considers three different keys. And some are function-based policies, such as LRV [15], GDSIZE [16] and etc, which uses a cost function to evaluate the data items. Compared with the three basic algorithms, other algorithms are generally more complicated variations and/or combinations of the three fundamental algorithms. For this reason, we choose the three basic algorithms LRU, LFU and SIZE as our evaluation algorithms.

### 3 Replacement Algorithms in Caching Hierarchy

#### 3.1 Problem Model for the Caching Hierarchy

For simplicity, we only consider a two-level hierarchical caching model as shown in Fig. 1.



**Fig. 1.** The basic structure of the problem model

We try to identify the working pattern and relationships between the two different cache layers, so that we may find a way to achieve a good performance for the two layers as a whole. The same knowledge can be extended to more complex hierarchies such as 4 or 5 different layers of cache.

To investigate the performance for the problem model, some issues need to be considered here:

- (1) The *request pattern* for different layers can vary. The lower-level caches have two kinds of ‘effects’ on the upper-level caches, which may greatly affect the requesting pattern. One is the ‘filter effect’, which means that only the requests missed at the lower levels can reach caches at upper levels. Another is the ‘aggregation effect’, as the requests to an upper-level cache are usually aggregated from multiple lower-level caches. Owing to these two effects, some original features about the request pattern, such as temporal locality and specific personal preferences, may be lost at upper levels.

- (2) The *size* and *capability* of the cache may be different for different layers. The sizes of the upper-level caches are usually larger, and their processing capabilities are normally stronger. It is suggested that the caches at different levels deal with the files with different sizes, i.e., large files are processed at the upper levels and small files are processed at the lower level [5].

In order to achieve a good performance for the integrated caching hierarchy, we need to consider different situations for different layers and find suitable algorithms.

### 3.2 Performance Metrics

Among all the different performance metrics for replacement algorithms, hit ratio (HR, or document hit ratio) is the most basic and important one. However, using HR assumes that data items in the cache all have the same size. After the cache has been utilized in the network environment, the sizes of the cached items become variable. As a result, BHR (byte hit ratio) is used as a performance metric. In the recent days, some more specific factors, such as the latency time and the traffic cost, are taken into account. In the following discussion, we will use both the HR and BHR as the basic criteria for the replacement algorithms' performance analysis. In most situations, the HR and BHR can also give a reasonable estimation for other performance metrics such as the total latency and traffic load.

## 4 Experiment

### 4.1 Experiment Design

We use trace-driven simulation to evaluate the performance of the various replacement algorithms in the caching hierarchy. The evaluation is performed based on three fundamental algorithms, LRU, LFU and SIZE, which are considering the three most basic factors for the replacement algorithm deployment: *ATIME* (last access time), *NREF* (number of references) and *SIZE* (size of the data item).

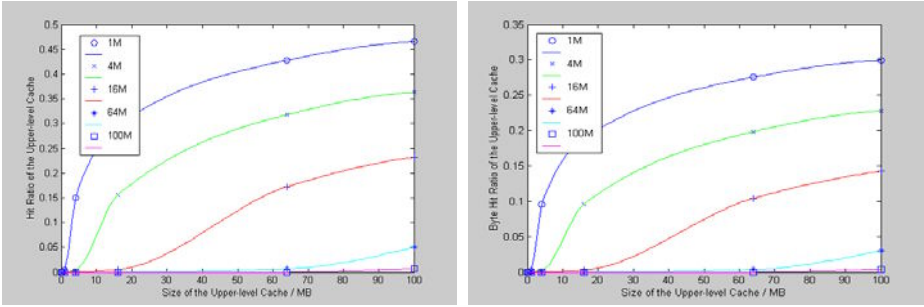
For the experiment data, we use the web proxy trace provided by the IRCache [2] website. The experiments are divided into two parts. First part uses uniform algorithms for different levels, but different cache sizes are used. Second part chooses different algorithms for different layers to achieve a better performance of the two caching layers. In both parts, we are trying to identify the characteristics of these algorithms, which may serve as the guidance to improve the performance.

For the cache sizes, we choose five different values for the upper-level cache and the lower-level caches (the sizes of lower-level caches are counted together as one value): 1MB, 4MB, 16MB, 64MB, and 100MB. The sizes of the upper-level cache and the lower-level may vary with the time. The details will be discussed in the following parts.

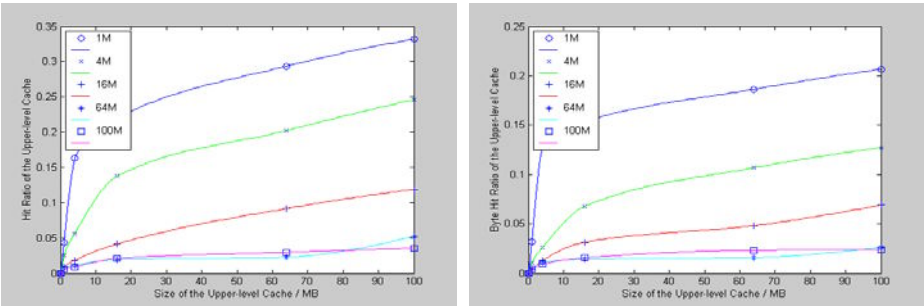
### 4.2 Individual Investigation into Three Basic Algorithms: LRU, LFU, and SIZE

This is the first part of the experiment, in which we tested the performance of the three basic algorithms in the cache hierarchy situation separately. The upper level and

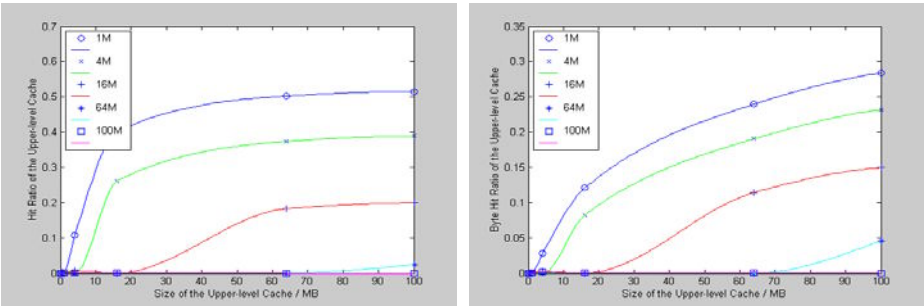




**Fig. 2.** Performance (HR/BHR) of the Upper-level Cache with LRU. The one on the left shows the HR while the one the right shows the BHR('16M' in the legend means the lower-level cache size is 16M and the curve shows how the performance of upper-level cache varies with upper-level cache size.)



**Fig. 3.** Performance (HR/BHR) of the Upper-level Cache with LFU



**Fig. 4.** Performance (HR/BHR) of the Upper-level Cache with SIZE

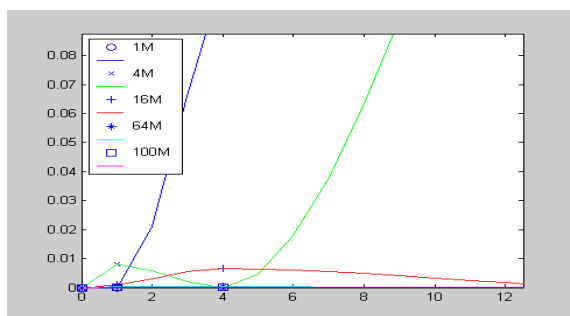
the lower level apply the same algorithm. We will observe how these algorithms perform in such situations.

The size of the upper-level cache is assigned with five different values: 1MB, 4MB, 16MB, 64MB and 100MB. The lower-level cache size varies from 1MB to 100MB, with also the 5 different values. So there will be totally 25 different cases for each algorithm. In the above figures 2, 3 and 4, for each curve, the lower-level cache size is a fixed value, and the curves exhibit how the upper-level cache size affects the cache performance.

For the performance of the upper-level cache, we observe a very clear ‘filter-effect’ from the lower-level caches. Let the size of the lower-level caches be  $X$ , if the size of the upper-level cache is equal to or less than  $X$ , the performance of the upper-level cache would be very poor (the HR and BHR would be very low, nearly 0). Only after the size of the upper-level cache has exceeded the size of the lower-level caches, the upper-level cache shows a better performance.

In the following, we identify some different characteristics among these three algorithms:

- (1) Although all these three algorithms perform poorly at the upper-level cache, LFU is regarded as the one to have the best performance. Let the size of the lower-level caches be value  $X$ . For LRU and SIZE, if the size of the upper-level cache is equal to or less than  $X$ , the HR will be around the value of 0.1%. If the lower-level cache size is too large, the HR will be less than 0.01%. While for LFU, even if the size of the upper-level cache is equal to or less than  $X$ , the HR will be more than 1%. Thus LFU is more resistible to the filter-effect of the lower-level cache and performs better in the caching hierarchy.
- (2) However, although LFU performs better when the upper-level cache size is less than lower-level cache, when the upper-level cache size has exceeded the lower-level cache size, the performances of LRU and SIZE improve faster than that of LFU with the increase of the upper-level cache size. So when the upper-level cache size is very large, the performances of LRU and SIZE are better at the upper level.
- (3) Opposite to LFU, SIZE is regarded to have the worst performance in the cache hierarchy. This is because SIZE strongly depends on the size of the cache. Assume the size of the lower-level cache to be  $X$ . If the size of the upper-level cache is less than  $X$ , the value of HR and BHR of the upper-level cache is very low, but still above zero. However, if the size of the upper-level cache is equal to  $X$ , the values of HR and BHR drop to zero.



**Fig. 5.** Unusual Pattern of the SIZE algorithm

This unusual pattern is shown in Fig. 5, which is a magnified part of Fig. 4. In this figure, we could see that each curve would drop to zero again when the upper-level cache size becomes the same as lower-level cache size. And the unusual pattern of the SIZE algorithm shows that the working pattern of SIZE strongly depends on the size of the cache.

Theoretically speaking, the replacement algorithms should check all the possible data items to see their probability to be accessed again in the near future. However,

for practical applications, it is just impossible to evaluate all the possible data items, and only the data items stored in the cache would be evaluated. In this way, the cache size, which determines how many data items are stored and evaluated for the replacement, would also be an important factor to determine the working pattern. So for the SIZE algorithm, when the upper-level and lower-level cache sizes are the same, the caches at two levels work in the same pattern. The lower-level caches simply filter out all the possible hits of the upper-level cache. In this kind of situation, keeping the sizes at different levels different could also be helpful to the overall performance.

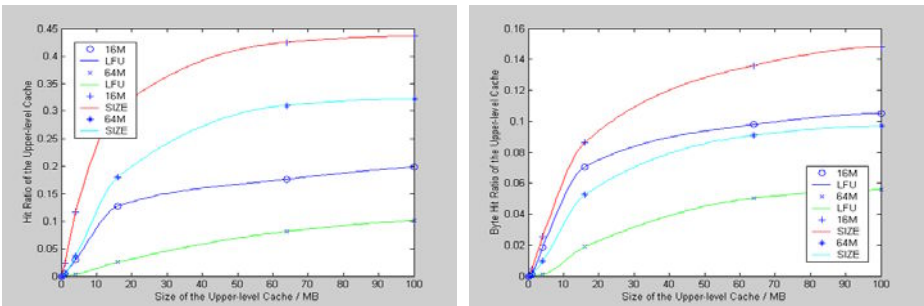
### 4.3 Different Algorithms at Different Levels

In the second part of the experiment, we use different algorithms at different levels. We try all the possible combinations of the three basic algorithms at different levels, hoping to identify the efficient working pattern of replacement algorithms in caching hierarchy.

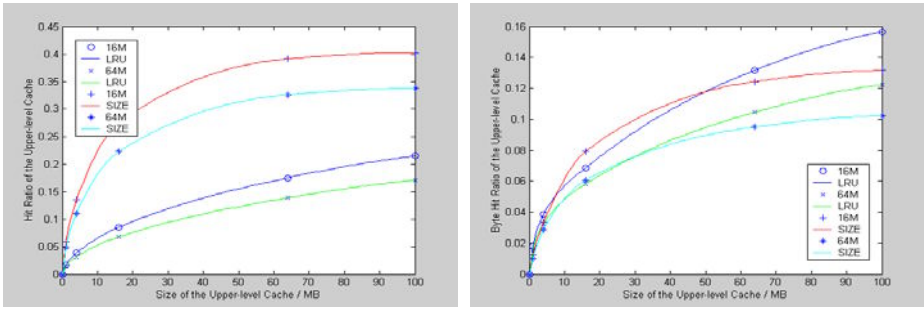
The cache sizes at the upper-level vary from 1MB to 100MB. But for the lower-level cache, the size only changes between 16MB and 64MB. This is because the filter-effect of the lower-level cache with a size of 1MB, 4MB or 100MB will either be too weak or too strong for us to measure the performance of the upper-level cache. Besides, in real environment, we don't expect the cache sizes of two different levels will be 100 times different. Then two different size values for the lower-level cache may be enough. The following figures show the performance of different algorithm combinations.

From these figures, we can clearly see that the performance of the upper-level cache is much better with different algorithms at different levels. Even when the size of the upper-level cache is smaller than the lower-level cache, the upper-level cache still gains a fairly good value of HR and BHR. And the filter-effect is almost 'gone'.

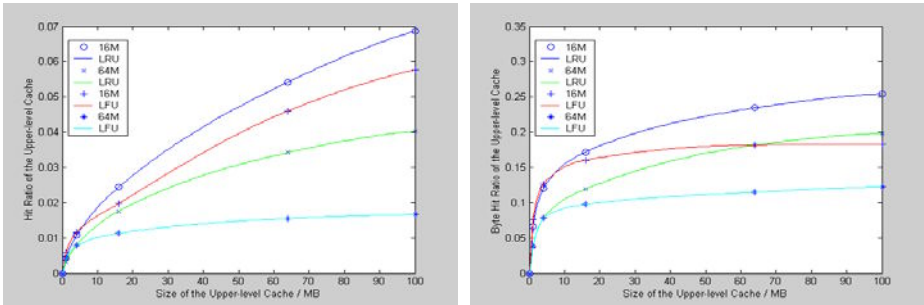
However, for different algorithm combinations, the performance is quite different. Fig. 6 shows that, when LRU is used as the lower-level cache algorithm, both HR and BHR of SIZE are much higher than that of LFU at the upper level. Fig. 7 demonstrates that, when LFU is used as the lower-level cache algorithm, the HR of SIZE is higher than that of LRU. For the BHR, when the upper-level cache size is small, the BHR of SIZE will be higher than or equal to LRU. But when the size becomes very large, the BHR values of LRU are higher. In Fig. 8, for which SIZE is used as the lower-level cache algorithm, the HR and BHR of LRU is much higher than LFU.



**Fig. 6.** Performance (HR/BHR) of Upper-level Cache with LRU as the Lower-level Cache Algorithm ('16M LFU' in the legend means the lower-level cache size is 16M and the upper-level cache is using LFU as the replacement algorithm.)



**Fig. 7.** Performance (HR/BHR) of the Upper-level Cache with LFU as the Lower-level Cache Algorithm



**Fig. 8.** Performance (HR/BHR) of the Upper-level Cache with SIZE as the Lower-level Cache Algorithm

In general, we observe that the performance of SIZE combining with either LRU or LFU is fairly better than that of LFU working with LRU. The combination of SIZE and LRU is even better than SIZE with LFU in terms of the performance.

So far the performances of the upper-level cache and the lower-level caches are evaluated separately. We need a method to evaluate the performance of the caching hierarchy as a whole. Denote the total HR and BHR of the lower-level caches as  $hr_L$  and  $bhr_L$ , and the HR and BHR of the upper-level cache as  $hr_H$  and  $bhr_H$ . We model the approximate evaluation of the whole hierarchy as ' $hr = hr_L + (1 - hr_L) * hr_H$ ' and ' $bhr = bhr_L + (1 - bhr_L) * bhr_H$ ', where  $(1 - hr_L) * hr_H$  and  $(1 - bhr_L) * bhr_H$  are the performances of the upper level. However, this is only an approximate indication of the performance, as the hits at the upper-level and the hits at the lower-level are still different in effects on time latency and traffic cost.

**Table 1.** Performance of the Caching Hierarchy as a Whole ('A-B' in the table means that A is used as upper-level algorithm and B is used as lower-level algorithm). Upper-level cache size is 16MB, and lower-level cache size is 64MB

Algorithms	LFU-LRU	SIZE-LRU	LRU-LFU	SIZE-LFU	LRU-SIZE	LFU-SIZE
HR	56.53%	63.42%	57.04%	64.20%	68.85%	68.65%
BHR	37.77%	39.91%	38.11%	38.27%	38.91%	37.42%

The above table shows the gross performance of the caching hierarchy with different algorithm combinations in a typical case (the results of other cases show the same comparison between different combinations). From the table, we could see that combination of SIZE and LRU produces the highest performance. And we suggest that SIZE be used as the lower-level algorithm, because it is good at dealing with the small-sized files and it is better to let the lower-level cache handle the small files.

## 5 Conclusions

We have done a thorough investigation for the performance of replacement algorithms in the caching hierarchy, using three most basic replacement algorithms (LRU, LFU and SIZE), which consider the three most basic factors (ATIME, NREF and SIZE) for cache replacement.

We first apply the same algorithms at both the upper and lower layer to see their performances in the caching hierarchy environment. And we observed that ‘filter-effect’ has caused the poor performance of the upper-level cache. Different algorithms demonstrated different characteristics in our simulations. The LFU algorithm performs the best in the hierarchy situation. But when the upper-level cache size becomes very large and the filter-effect of the lower-level cache is no longer an important factor, the LRU and SIZE outperforms LFU. Another special characteristic we have found is about the SIZE algorithm. Its working pattern has a strong dependence on the size of the cache. If the cache sizes of the two levels are the same, the HR and BHR of the upper level will fall to zero.

For the second part of our simulation, we have tested the performance of the combinations of different algorithms. We use different algorithms for caches at different levels. In this case, the ‘filter-effect’ problem can be almost solved. Even if the upper-level cache size is smaller than the lower-level cache size, the upper level could still provide an acceptable performance. We also compare the performance of these different combinations of algorithms. The conclusion is that combining SIZE with either LRU or LFU would produce better performance than combining LRU and LFU. And between the two, SIZE working with LRU will be better than SIZE with LFU. Based on extensive experiment results, we found that the combination of SIZE at the lower level and LRU at the upper level achieve the highest performance among all these combinations.

The following issues can be further investigated: (1) how to evaluate the performance of a caching hierarchy as whole precisely, which we have mentioned in the experiment part when we try to evaluate the gross performance of different algorithm combinations. (2) Pre-fetch scheme: We would like to see whether the technique of pre-fetch could be used in the caching hierarchy to improve the caching performance.

## References

1. K. Claffy and H.-W. Braun, “Web traffic characterization: An assessment of the impact of caching documents from NCSA’s web server”, Proc. 2nd World Wide Web Conf. Mosaic and the Web, 1994.

2. IRCache Homepage: <http://www.ircache.net>.
3. N. G. Smith, "The UK national Web cache—the state of the art," in *Proc. Computer Networks and ISDN Systems*, vol. 28, 1996, pp. 1407–1414.
4. JWCS Homepage: <http://www.cache.ja.net/index.html>.
5. M. Busari and C. Williamson, "Simulation Evaluation of a Heterogeneous Web Proxy Caching Hierarchy", *Proceedings of MASCOTS*, pp. 379-388, Cincinnati, OH, August 2001.
6. R. Tewari, M. Dahlin, H. Vin and J. Kay, "Beyond Hierarchies: Design Considerations for the Distributed Caching on the Internet", *Proceedings of the 19th International Conference on Distributed Computing Systems*, Austin, TX, June 1999.
7. D. Povey and J. Harrison, "A Distributed Internet Cache", *Proceedings of the 20th Australian Computer Science Conference*, Sydney, Australia, February 1997.
8. V. Valloppillil and K. Ross, "Cache Array Routing Protocol v1.1", Internet Draft, February 1998.
9. P. Yu and E. MacNair, "Performance Study of a Collaborative Method for Hierarchical Caching in Proxy Servers", *Proceedings of World-Wide Web Conference*, pp. 215-224, April 1998.
10. L. Fan, P. Cao, J. Almeida and A. Broder, "Summary cache: A scalable wide-area web cache sharing protocol," in *Proc. SIGCOMM'98*, Feb. 1998, pp. 254–265.
11. C. Aggarwal, Joel L. Wolf and P. S. Yu, "Caching on the World Wide Web", *IEEE Transactions on Knowledge and Data Engineering*, vol. 11, no. 1, January/February 1999.
12. M. Abrams, C. R. Standridge, G. Abdulla, S. Williams and E. A. Fox, "Caching proxies: Limitations and potentials", 4th International World-wide Web Conference, pages 119-133, Dec, 1995.
13. M. Arlitt, L. Cherkasova, J. Dilley, R. Friedrich, T. Jin, "Evaluating Content Management Techniques for Web Proxy Caches", *Proceedings of the Workshop on Internet Server Performance (WISP99)*.
14. S. Williams, M. Abrams, C. R. Standridge, G. Abdulla and E. A. Fox, "Removal Policies in Network Caches for World-Wide Web Documents", *Proceedings of ACM SIGCOMM*, pp. 293-305, 1996.
15. L. Rizzo and L. Vicisano, "Replacement Policies for a Proxy Cache", *IEEE/ACM TRANSACTIONS ON NETWORKING*, VOL. 8, No. 2, APRIL 2000.
16. P. Cao and S. Irani, "GreedyDual-size: A cost-aware WWW proxy caching algorithm", in *Proc. 2nd Web Caching Workshop*, Boulder, CO, June, 1997.

# Component-Based WebGIS and Its Spatial Cache Framework

Yingwei Luo, Xiaolin Wang, and Zhuoqun Xu

Dept. of Computer Science and Technology, Peking University, Beijing, 100871, P.R.China  
lyw@pku.edu.cn

**Abstract.** Component model is a primary approach to deepen the functions of WebGIS. A component-based WebGIS system Geo-Union is introduced. Geo-Union has a multi-level Client/Server architecture, including four layers: application layer, component layer, service layer and storage layer, where service layer has different units to provide both client services and server services. Also, component partition and web mode of Geo-Union is discussed. After that, a spatial cache framework, including spatial database cache, network spatial cache and spatial data proxy server, is designed in Geo-Union to improve spatial data access performance in different situations in network environment.

## 1 Introduction

Internet is changing the means of data access and release. WebGIS, which processes geographical spatial data in Internet, is developing rapidly, along with the movement of Internet and Web technique. Just because of the development and application, Internet should become the running platform of further GIS, and Web-based GIS applications are new urgent demands of most GIS users.

Component affords a new software construction mode, which is more efficient, agile and powerful. Developers could combine and reuse binary modules that independently exploited by different individual or groups. Component is independent of language and hardware, and can run on Web. Moreover, it provides much more flexibility for application development<sup>[1][2]</sup>.

Right now, there are too many WebGIS products in market, mainly including: Internet Map Server (IMS) and MapObject from ESRI, the former is used to release map on Web, and the last is a component library for further development of GIS applications<sup>[3]</sup>; MapInfo ProServer and MapX from MapInfo, their capabilities are similar to ESRI's IMS and MapObject<sup>[4]</sup>; MapGuide from Autodesk<sup>[5]</sup>; GeoMedia Web Map from Intergraph<sup>[6]</sup>, GeoSurf from GeoStar<sup>[7]</sup> and so on. Those WebGIS products are different from their own design and implementation, but their basic thought is the same. For the implementation techniques, all of them adopt CGI/Serverlet technique, Plug-in technique, CORBA/DCOM technique, Java Applet technique and so on, or integrate several above techniques. For the implementation mode, there are two models. The first is thin client model. At server side, spatial data

is mainly in vector format and most functions are completed there; but at client side, results are displayed in image. In thin client model, each operation of users must interact with server, so there are too many conversations and data transmissions between client and server. The second is fat client model. Spatial data is transferred from server to client in vector format, and most functions are complete in client. In fat client model, computing capability in client is uncertain and there exist many network security leaks, so most WebGIS products only implement some basic functions, such as elementary map visualization and simple spatial query.

Assigning functions in reason and improving performance are two key issues for making WebGIS more practicable. In fact, some WebGIS products, especially component-based WebGIS products, have already achieved a high practicability. But for a market reason, their design and implementation technologies are not open. So at first, we analyze the modeling technique of component-based WebGIS, construct a practicable, multi-level WebGIS system Geo-Union, and explore its architecture, composition and functional partition of components. Also, a spatial cache framework used to improve performance in Geo-Union is given.

## 2 Geo-Union: Architecture, Component Partition and Web Mode

### 2.1 Architecture

Component model is a primary approach to deepen the functions of WebGIS. Geo-Union is divided into four layers: application layer, component layer, service layer and storage layer, where service layer has different units to provide both client services and server services. Figure1 shows the architecture <sup>[8][9]</sup>. Hierarchical spatial component object model can distribute GIS functions in network reasonably and make the system reusable, as well as provide efficiency approach for further development and integration with other systems.

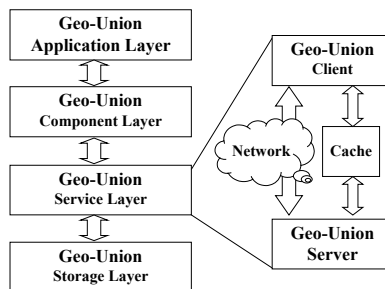


Fig. 1. Geo-Union Architecture

- (1) Storage layer is responsible for storage and management of both spatial and non-spatial data based on ORDB. The main problems solved at this layer are how to represent and store spatial data, and how to maintain relationships among spatial data.



- (2) Service layer is in charge of spatial data access and process, and includes two parts: Geo-Union client provides data access and process services to component layer, and Geo-Union server provides data access and process services to Geo-Union client through interacting with storage layer. Geo-Union server can manage different spatial data resources, and also can reply to different spatial data requests from different clients. Geo-Union server provides the services of data access, spatial index, basic spatial query, transaction process, data share and so on. Geo-Union client provides different GIS tools and further development functions to component layer based on the services from Geo-Union server. Cache is an important unit of service layer, which is imported to reduce network load and improve response speed of the system. Using Geo-Union client, we can develop a simulation server, which can reduce network load and improve response speed through its cache (see section 3).
- (3) Component layer provides a rich set of services (components) to develop domain-oriented GIS applications for further developers. Component layer provides interface of GIS functions to users, but the implementation details are completed in service layer. Component layer exists as a component library, and servers as a bridge between users and service layer. Component layer provides function-explicit and reusable interface components for users according to the functions of service layer.
- (4) The work in application layer is to exploit application systems for different special domains by assembling and integrating Geo-Union components. These application systems can be running both in desktop and network environment.

## 2.2 Component Partition

Component-based WebGIS makes it possible that application developers, data promulgators and spatial database engine vendors construct domain-oriented GIS applications together. Moreover, domain-oriented GIS components can be built conveniently according to specified models, and putted into component library at any time, thus used by other users. All of these will make the system possess high reusability.

Most of functions are implemented in service layer. In order to get high efficiency of reusability and high flexibility of assembly, it is important to give a clear functional partition and design a right architecture for service layer.

Because the function of server side is just to provide spatial data access, so Geo-Union server serves as an application, not as any components. Geo-Union server can manage different spatial data resources, and also can reply different spatial data requests from different clients.

But Geo-Union client supports to develop domain-oriented applications. In order to provide flexible developing mode, around map visualization objects, we designed another six types of objects: spatial data access objects, map edit objects, spatial analysis objects, mouse tool objects, utility objects and AppTool. Figure 2 shows the architecture of service layer, the roles of different objects and their relationships<sup>[8][9]</sup>.

Spatial data access objects connect with Geo-Union server and access spatial data stored in spatial database. GxConnection is the core object of spatial data

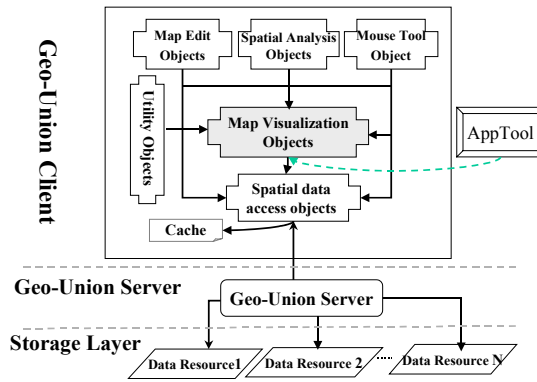


Fig. 2. Architecture and Functional Partition of Service Layer

access objects. Through GxConnection, we can perform spatial data manipulating operations such as opening, adding, deleting, updating and querying corresponding data in spatial database. Also, a special series of objects are designed to provide spatial cache mechanism (see section 3).

Map edit objects encapsulate mouse actions to edit spatial entity, such as inputting a point (GxInputPoint), a line (GxInputLine) and a polygon (GxInputPolygon).

Spatial analysis objects provide spatial analysis functions such as overlay analysis (GxOverlay), buffer analysis (GxBuffer) and network analysis (GxNetwork).

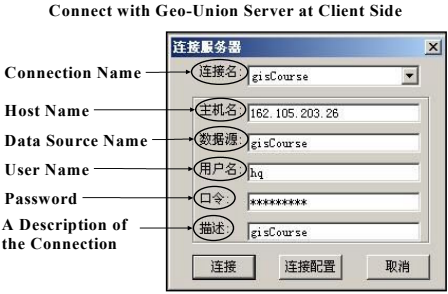
Mouse tool objects are similar to map edit objects. They also encapsulate and manage mouse actions to form some basic GIS operations, such as zooming in map (GxZoomIn), zooming out map (GxZoomOut), roaming in map (GxPalm) and picking entity from map (GxPick).

Utility objects provide an object factory (GxFactory) and some data structure objects such as array (GxArray), set (GxSet) and enumeration (GxEnumeration). Utility objects are frequently used but may not be easily defined in some development environments (such as Visual Basic and VBScript). For example, in VBScript, some programmable objects cannot be declared directly, but they can be easily generated by GxFactory. Also, Utility objects contain a GxError to provide an error mechanism.

Map visualization objects are directly used to organize spatial data (layer and map) and display them. With map visualization objects, we can set a map to be displayed and the display attributes of the map, such as display scale and scope of the map. Meanwhile, map visualization objects can be combined with other objects to provide various additional visualization functions, for example, map zooming, map roaming, entity picking, spatial analysis result displaying, etc. Furthermore, there include some other functions such as dynamic layer displaying (GPS tracker) and map outputting. Map visualization objects contain a map visualization control (GxMapView) and three programmable objects: map object (GxMap), display layer object (GxMapLayer) and dynamic layer object (GxTrackingLayer)<sup>[8][9]</sup>.

AppTool is an application-oriented component that integrates more GIS operations and user's interfaces using above objects. For example, *AppTool.Connect()*, a

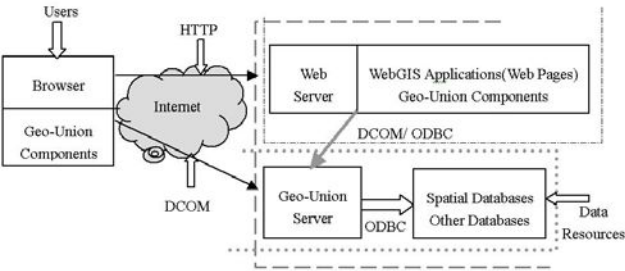
method of AppTool, provides a dialog for user or further developer at client side to indicate remote host, spatial data source, login user name and password when connecting with Geo-Union server (As shown in figure 3). This method integrates GxConnection in data access objects.



**Fig. 3.** An Example of AppTool – AppTool.connect()

### 2.3 Web Mode

Actually, Geo-Union component layer is a set of ActiveX controls and correlative programmable objects. ActiveX controls can be embedded into Web pages directly and programmable objects also can be used through ASP, by means of which Web browser and GIS combine with each other. Figure 4 shows the Web application mode.



**Fig. 4.** Web Application of Geo-Union

Users betake WebGIS applications through browser, and browser can attain geographical spatial services by direct mode or indirect mode.

Direct mode means browser use Geo-Union components directly at client side. WebGIS applications are downloaded from Web server and execute at client side with the help of Geo-Union components. Required spatial data is transferred from Geo-Union server to client, and users' requests are processed at client side.

Indirect mode means users' requests are submitted to Web server, and WebGIS applications are execute at Web server side with the help of Geo-Union components to process the requests. Required spatial data is transferred from Geo-Union server to

Web server. Finally, a dynamic Web page that reflects the result of the requests is returned to browser.

In both modes, Geo-Union component layer is the indispensable pivot in whole system.

### 3 Spatial Cache Framework

Cache is an important technique for improving the performance of system. In Geo-Union, there are two aspects affecting the efficiency of data access: one is the access to database, especially when storing spatial data in ORDB; the other is the transmission of spatial data in network. We take different spatial cache modes to solve these two problems in Geo-Union.

#### 3.1 Spatial Cache Framework

Figure 5 shows the spatial cache framework in Geo-Union, which includes three typical cache modes: database cache, network cache and data proxy server.

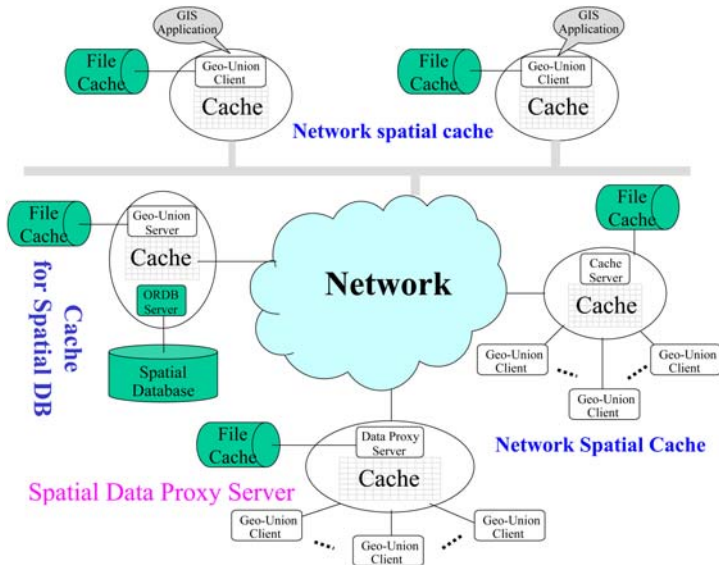


Fig. 5. Spatial Cache Framework

##### (1) Cache for spatial database

Geo-Union server is the bridge between Geo-Union client and ORDB, so cache for spatial database is established and maintained by Geo-Union server. Cache is stored in both local file and memory. Because Geo-Union server can manage different spatial data resources, and also can reply different spatial data requests from different clients, so cache for spatial database is a global cache.

It is definite that retrieving a group spatial entities from ORDB, especially some neighboring entities in a layer, is much more inefficient than file-structure based GIS. Aiming to this difference, ORDB – memory file cache mechanism is adopted in Geo-Union to speed up spatial entity access. Here memory file is opened in memory and can be random accessed like memory page. If memory in spatial database server is great enough, memory file cache mechanism is more effective than pure file system.

## (2) Network spatial cache

The bandwidth of different users in network is different, but public users of Internet always have low bandwidth. So when people use WebGIS applications, it is a lethal delay when spatial data is transferred from Geo-Union server. In Geo-Union client, we adopt a two-level spatial cache mode to relief transmission bottleneck of spatial data in the network:

The first network cache is used to help a single client access remote data: building a spatial cache in local place. This kind of cache is a partial cache, which is also a popular method in today's Web browser.

The second network cache is used to help many clients in a LAN access remote data: building a common spatial cache for a LAN (building a cache server). Once a local client in the LAN accessed some spatial data, other clients can reuse the spatial data in cache server. Cache server is still a partial cache.

Cache server can realize massive spatial data cache by means of the shared resources, which will speed up the hit rate of cache greatly, so as to improve efficiency of all local clients, and save resource of all local clients. Cache server solves the speed conflict between local disk data access and remote data access, and the speed conflict between the high-speed LAN and WAN with narrow bandwidth.

## (3) Spatial data proxy server

Because the distribution of users in Internet is not well proportioned, so different Geo-Union servers are unbalanced. Some Geo-Union server and its communication may overload. Aiming to this problem, we design spatial data proxy server for those Geo-Union servers to improve performance.

Spatial data proxy server is an initiative cache server. Overloaded Geo-Union server selects a suitable Geo-Union client and builds a spatial data proxy server there, cache all or part of spatial data in spatial database, and response a special group of users.

In Geo-Union, spatial data proxy server serves as special server in Internet to provide spatial data access services for public users. The structure and implementation of spatial data proxy server is same as cache server, but they play a different role in Geo-Union.

Cache server is private of a LAN, and clients in the LAN have to get an authorization before accessing to cache server. Spatial data in cache server is changing with different requests of clients in the LAN.

Spatial data proxy server is public to all clients. Spatial data proxy server may serve as a peer of Geo-Union server. Spatial data proxy server can be built anywhere in Internet if needed. If spatial data in a spatial database is unchanged for a long time, spatial data proxy server can cache all spatial data of that spatial database.

Building spatial data proxy servers properly in Internet will make Geo-Union applications more scalable and effective.

### 3.2 Organization of Spatial Cache

In Geo-Union, spatial cache is organized as three levels: layer, slot and entity.

When creating a layer in spatial database, a GUID (Global Universal Identification) is generated to identify the layer, which is named as *layerID*. When reading a layer into cache, the system will allocate a separate space for the layer according to its *layerID*. Whether a layer is valid in cache is determined by the *layerVersion* of the layer both in cache and in spatial database.

Entities in a layer are always separated into different slots according to a certain rule. When reading a layer into cache, we do not read the whole layer, but read some slots of the layer. Slot brings two benefits: almost all spatial queries do not need a whole layer but only a certain scope in the layer, so when the layer is massive, reading relative slots can satisfy the requirement and will reduce network load greatly; less data will exhaust less computing resource and storage resource. The rules to organize slot are various. We can organize slots in a layer according to a correlativity of geographical location or a neighborhood geographical location. A correlativity of geographical location means we can put entities along a railway into a slot, and a neighborhood geographical location means we will put entities in a certain spatial scope into a slot. Every entity in a layer belongs to a slot. When entities in any slots changed, the *slotVersion* of the layer will change too.

One update operation may modify only one or several entities in a layer, so *layerVersion* and *slotVersion* of the layer cannot reflect the latest modification of entities. We set a *versionNumber* for every entity, and when an entity changes, its *versionNumber* changes too. The *entityVersion* of a layer is the max *versionNumber* in the layer. When the *versionNumber* of a layer in cache is less than that in spatial database, some entities in cache is invalid and those entities that have larger *versionNumber* should be reloaded from spatial database into cache.

### 3.3 Refresh and Pre-load Spatial Cache

Refreshment of spatial cache can be done online or offline. Online refreshment means updating spatial data in cache at the same time as updating spatial entities in a layer in spatial database. Offline refreshment means updating invalid spatial data of a layer in cache if their versions are invalid when accessing to them.

Information in spatial cache includes *layerVersion*, *slotVersion*, *entityVersion* and the corresponding relations between all entities and slots of a layer. When accessing to entities in a layer, we can determine whether spatial data in cache is valid or not through comparing version information in cache and spatial database.

- (1) If *layerVersion* of a layer in cache is smaller than that in spatial database, it means that all entities of the layer are changed, and the whole layer should be refreshed in cache. Otherwise,

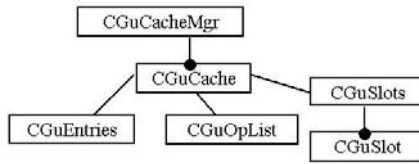
- (2) If *slotVersion* of a layer in cache is smaller than that in spatial database, it means that all slots of the layer are rearranged, and the corresponding relations between all entities and slots of the layer should be refreshed in cache. Otherwise,
- (3) If *entityVersion* of a layer in cache is smaller than that in spatial database, it means that some entities of the layer are changed, and those entities should be refreshed in cache.
- (4) If above three conditions are all equal to those in spatial database, it means spatial data of a layer in cache is same as that in spatial database, and no refreshment is required.

When accessing to spatial data, if we can pre-load some spatial data into cache, it will make spatial cache more effective. But how to predict what kind of spatial data will be accessed to?

There are two rules of accessing to memory: there is every probability of accessing to just being accessed memory, and there is every probability of accessing to neighbors of just being accessed memory. There are same rules in accessing to spatial data: there is every probability of accessing to just being accessed spatial data, and there is every probability of accessing to neighbors of just being accessed spatial entities. According to those rules, we can arrange slots by spatial scope, and a slot is an accessing unit of spatial data. When the network is idle, we can pre-load some neighbors of spatial data in cache from spatial database.

### 3.4 Spatial Cache Object

In Geo-Union, different spatial caches are all implemented by the six spatial cache objects: CGuCacheMgr, CGuCache, CGuSlots, CGuSlot, CGuEntries and CGuOpList. Figure 6 illustrates the hierarchies of them.



**Fig. 6.** Hierarchies of Spatial Cache Objects

In every spatial cache, there is only one CGuCacheMgr object, which manages all CGuCache objects. A layer in cache is represented by a CGuCache object. When opening a layer, CGuCacheMgr object will create a CGuCache object or find an existing CGuCache object through its *layerID*.

CGuEntries object is used to manage a mapping list between an *entityID* and entity data of a layer. From the mapping list, we can know the *slotID* where an entity belongs to (if *slotID* = 0, it means an entity is not existing; if *slotID* > 0, it means an entity is in cache; and if *slotID* < 0, it means an entity is not in cache), and the address pointer of the entity data if the entity is in cache.

CGuOpList object is used to record updating operation list of a layer at client side that still doesn't be committed to spatial database. When the updating operation list is committed, it will also update the corresponding spatial data in cache. But when the updating operation list is cancelled, no change will happen in cache.

## 4 Conclusion

Geo-Union has finished a preliminary component-based model for distributed WebGIS, and has got into use in many fields with sound effects. Although spatial cache and other techniques are adopted in Geo-Union to improve its performance, a lot of works still wait us ahead to make Geo-Union more practicable and effective:

- (1) Dynamic load balancing policy. In Geo-Union, most works are completed at client side, and server is only responsible for data access and simple data query. Therefore it is not well balanced between client and server. Especially there will exists a lot of transmission for massive data between client and server. Although, spatial index and spatial cache techniques can improve the performance to a certain extent, we still want to take full advantage of the computing capability of GIS server, so as to lighten load at client side and decrease the transmission of redundant data in network. This needs a more reasonable component design for the system.
- (2) System concurrency. WebGIS is open to millions of users. How to ensure the correctness, validity, stability and scalability of Geo-Union to satisfy users' requests is another key problem for practicable WebGIS.
- (3) System intelligence. Agent technique is a development trend and provides a new model for in distributed software construction. Of course, agent technique will bring a new thought for distributed GIS<sup>[10]</sup>. How to apply agent technique into distributed GIS, and provide interoperable services, collaborative services and intelligent services with the help of spatial metadata, are most important and significant researches in distributed GIS.

## Acknowledgement

This work is supported by the National Research Foundation for the Doctoral Program of Higher Education of China (20020001015); the National Grand Fundamental Research 973 Program of China (2002CB312000); the National Science Foundation of China (60203002); the National High Technology Development 863 Program (2002AA135330, 2002AA134030); the Beijing Science Foundation (4012007).

## References

1. Li Bin: A Component Perspective on Geographic Information Services, Cartography and Geographic Information Science, 27(1): 75-86(2001).
2. Szyperiski C: Component software: beyond object-oriented programming Reading, MA: Addison-Wesley Press, 1998.



3. <http://www.esri.com>.
4. <http://www.mapinfo.com>.
5. <http://www.autodesk.com>.
6. <http://www.intergraph.com>.
7. <http://www.geostar.com.cn>.
8. Dept. of Computer Science and Technology: Peking University, Operation and Component Guide for Geo-Union Enterprise (in Chinese), Technology Material, <http://gis.pku.edu.cn>.
9. Li Muhua: Study on Component based WebGIS and Its implementation [Master Dissertation] (in Chinese), Beijing: Peking University, 2000.6.
10. <http://map.sdsu.edu/geoagent/>.

# Learning-Based Top-N Selection Query Evaluation over Relational Databases\*

Liang Zhu<sup>1</sup> and Weiyi Meng<sup>2</sup>

<sup>1</sup> School of Mathematics and Computer Science, Hebei University  
Baoding, Hebei 071002, China  
zhu@mail.hbu.edu.cn

<sup>2</sup> Department of Computer Science, State University of New York at Binghamton  
Binghamton, NY 13902, USA  
meng@cs.binghamton.edu

**Abstract.** A top- $N$  selection query against a relation is to find the  $N$  tuples that satisfy the query condition the best but not necessarily completely. In this paper, we propose a new method for evaluating top- $N$  selection queries against relational databases. This method employs a learning-based strategy. Initially, it finds and saves the optimal search spaces for a small number of random top- $N$  queries. The learned knowledge is then used to evaluate new queries. Extensive experiments are carried out to measure the performance of this strategy and the results indicate that it is highly competitive with existing techniques for both low-dimensional and high-dimensional data. Furthermore, the knowledge base can be updated based on new user queries to reflect new query patterns so that frequently submitted queries can be processed most efficiently.

## 1 Introduction

As pointed out in a number of recent papers [1, 4–7, 9, 10, 15], it is of great importance to find the  $N$  tuples in a database table that best satisfy a given user query, for some integer  $N$ . This is especially true for searching commercial products on the Web. For example, for a Web site that sells used-cars, the problem becomes finding the  $N$  best matching cars based on a given car description.

A top- $N$  selection query against a relation/table is to find the  $N$  tuples that satisfy the query condition the best but not necessarily completely. A simple solution to this problem is to retrieve all tuples in the relation, compute their *distances* with the query condition using a *distance function* and output the  $N$  tuples that have the smallest distances. The main problem with this solution is its poor efficiency, especially when the number of tuples of the relation is large. Finding efficient strategies to evaluate top- $N$  queries has been the primary focus of top- $N$  query research.

In this paper, we propose a new method for evaluating top- $N$  selection queries against a relation. The main difference between this method and existing ones is that it employs a learning-based strategy. A knowledge base is built initially by finding

---

\* This work is completed when the first author was a visitor at SUNY at Binghamton.

the optimal (i.e., the smallest possible) search spaces for a small number of random top- $N$  selection queries and by saving some related information for each optimal solution. This knowledge base is then used to derive search spaces for new top- $N$  selection queries. The initial knowledge base can be continuously updated while new top- $N$  queries are evaluated. Clearly, if a query has been submitted before and its optimal solution is stored in the knowledge base, then this query can be most efficiently processed. As a result, this method is most favorable for repeating queries. It is known that database queries usually follow a Zipfian distribution [2]. Therefore, being able to support frequently submitted queries well is important for the overall performance of the database system. What is attractive about this method, however, is that even in the absence of repeating queries, our method compares favorably to the best existing methods with comparable storage size for storing information needed for top- $N$  query evaluation. In addition, this method is not sensitive to the dimensionality of the data and it works well for both low-dimensional and high-dimensional data.

The rest of the paper is organized as follows. In Section 2, we briefly review some related works and compare our method with the existing ones. In Section 3, we introduce some notations. In Section 4, we present our learning-based top- $N$  query evaluation strategy. In Section 5, we present the experimental results. Finally in Section 6, we conclude the paper.

## 2 Related Work

A large number of research works on the efficient evaluation of top- $N$  selection queries (or  $N$  nearest-neighbors queries) are reported recently [1, 4–7, 9, 10, 15]. Here we only review and compare with those that are most related to this paper.

The authors of [10] use a probabilistic approach to query optimization for returning the top- $N$  tuples for a given selection query. The ranking condition in [10] involves only a single attribute. In this paper, we deal with multi-attribute conditions.

In [6], histogram-based approaches are used to map a top- $N$  selection query into a traditional range selection query. In [1], the histogram-based strategies are extended to a new technique called *Dynamic*, expressed as  $dq(\alpha) = dRq + \alpha(dNRq - dRq)$ . A significant weakness of histogram-based approaches is that their performance deteriorates quickly when the number of dimensions of the data exceeds 3 [1, 14]. Therefore, histogram-based approaches are suitable for only low-dimensional data in practice.

In [7], a sampling-based method is proposed to translate a top- $N$  query into an approximate range query. Unlike histogram-based approaches, this method is suitable for high-dimensional data and is easy to implement in practice. However, this method only provides an approximate answer to a given top- $N$  query, i.e., it does not guarantee the retrieval of all of the top- $N$  tuples. In addition, for large databases, this method may be inefficient. For example, for a relation with 1 million tuples and using a 5% sample set (as reported in [7]), 50,000 tuples need to be evaluated in order to find the approximate range query for a top- $N$  query.

Our learning-based method is fundamentally different from all existing techniques. It can learn from either randomly generated training queries or real user queries so it can adapt to changes of query patterns. Furthermore, it delivers good performance for both low-dimensional and high-dimensional data.

### 3 Problem Definition

Let  $\mathfrak{R}^n$  be a metric space with distance (or, metric) function  $d(.,.)$ , where  $\mathfrak{R}$  is the real line. Suppose that  $R \subseteq \mathfrak{R}^n$  is a relation (or dataset) with  $M$  tuples and  $n$  attributes  $(A_1, \dots, A_n)$ . A tuple  $t \in R$  is denoted by  $t = (t_1, \dots, t_n)$ . Consider a point (query)  $Q = (q_1, \dots, q_n) \in \mathfrak{R}^n$ . A top- $N$  selection query, or top- $N$  query for short, is to find a sorted set of  $N$  tuples in  $R$  that are closest to  $Q$  according to the given distance function. The results of a top- $N$  query are called *top- $N$  tuples*.

Suppose  $Q = (q_1, \dots, q_n)$  is a top- $N$  query and  $r > 0$  is a real number. We use  $S(Q, r)$  to denote the  $n$ -square  $\prod_{i=1}^n [q_i - r, q_i + r]$  centered at  $Q$  with side length  $2r$ . Our goal is to find a *search distance*  $r$  as small as possible such that  $S(Q, r)$  contains the top- $N$  tuples of  $Q$  according to the given distance function. We use  $S(Q, r, N)$  to denote this smallest possible  $n$ -square and the corresponding  $r$  is called the optimal search distance for the top- $N$  query  $Q$ . Some example distance functions are [1, 7, 15]:

Summation distance (i.e.,  $L_1$ -norm or Manhattan distance):

$$d_1(x, y) = \sum_{i=1}^n |x_i - y_i|.$$

Euclidean distance (i.e.,  $L_2$ -norm distance):  $d_2(x, y) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2}$ .

Maximum distance (i.e.,  $L_\infty$ -norm distance):  $d_\infty(x, y) = \max_{1 \leq i \leq n} \{|x_i - y_i|\}$ .

Frequently, it is not suitable to apply a distance function to the values of different attributes directly due to the unit/scaling difference of the values in different attributes. For example, when buying a used car, a 100-dollar difference in price is probably more significant than a 100-mile difference in mileage. This can be resolved by multiplying appropriate importance/scaling factors to the raw distances based on the values of different attributes [9, 15]. Without loss of generality, in this paper, we assume that all attributes are of the same importance for ranking tuples so that the distance functions can be applied directly.

### 4 Learning-Based Top-N Query Evaluation

In this section, we introduce our learning-based method for top- $N$  query evaluation. First, keep track of frequently submitted queries and save the evaluation strategies for these queries in a knowledge base. Next, for each newly submitted query, if its

evaluation strategy has been saved or it is similar to some queries whose evaluation strategies have been saved, then derive an evaluation strategy for the new query from the saved relevant strategy/strategies. In this paper, we study issues related to this method.

#### 4.1 Query Information to Save

Initially, the knowledge base is empty. When a new query arrives at the system, an existing method such as the one proposed in [1] or in [7] is employed to evaluate the query. Let  $Q = (q_1, \dots, q_n)$  be the query and  $t_i = (t_{i1}, \dots, t_{in})$ ,  $1 \leq i \leq N$ , be the top- $N$  tuples, respectively, then the search distance of the smallest  $n$ -square is  $r = \max_{1 \leq i \leq N} \{d_\infty(Q, t_i)\} = \max_{1 \leq i \leq N} \{\max_{1 \leq j \leq n} \{|q_j - t_{ij}|\}\}$ . When the smallest  $n$ -square is obtained, several pieces of information are collected and saved. For a top- $N$  query  $Q$ , let  $r$  be the search distance of  $S(Q, r, N)$  – the smallest  $n$ -square that contains the top- $N$  tuples of  $Q$ ,  $f$  denote the frequency of  $S(Q, r, N)$ , i.e., the number of tuples in  $S(Q, r, N)$  (obviously  $N \leq f$ ),  $c$  denote the number of times that  $Q$  has been submitted, and  $d$  denote the most recent time when  $Q$  was submitted. The information that we will keep for each saved query  $Q$  is represented as  $\zeta(Q) = (Q, N, r, f, c, d)$  and it is called the *profile* of the query.

After the system has been used for sometime, a number of query profiles are created and saved in the knowledge base. Let  $P = \{\zeta_1, \zeta_2, \dots, \zeta_m\}$  denote the set of all query profiles, i.e., the knowledge base, maintained by the system. Queries in  $P$  will be called *profile queries*. Generally, profiles should be kept only for queries that are frequently submitted recently as reflected by the values of  $c$  and  $d$  in each profile.

In our implementation, the initial knowledge base is not built based on real user queries. Instead, it is based on randomly selected queries from a possible query space (see Section 5 for more details).

#### 4.2 New Query Evaluation

When a newly submitted top- $N$  query  $Q$  is received by the system, we need to find an appropriate search distance  $r$  for it. In a system that does not store query profiles, this query will be processed just like any new query and the methods discussed in [1, 7] may be used to find  $r$ . When query profiles are stored, it becomes possible to obtain the  $r$  for some new user queries from these profiles.

##### 4.2.1 Determining the Search Distance $r$

The details of obtaining the search distance  $r$  for a new query are described below.

First we identify  $Q' = (q'_1, q'_2, \dots, q'_n)$  from  $P$  that is the closest to  $Q$  under the distance function  $d(\dots)$ . The following cases exist.

1.  $d(Q, Q') = 0$ , i.e.,  $Q' = Q$ . In this case, find all profiles in  $P$  whose query point is  $Q'$ , but have different result size  $N$ , say,  $N_1 < N_2 < \dots < N_k$ . An example of a 2-dimension case is depicted in Figure 1, where squares of solid-lines represent the

search spaces of profile queries (i.e., those in  $P$ ) and the square of dotted-lines represents the search space of the new query  $Q$ . We now consider three subcases.

- a. There is  $N' \in \{N_1, N_2, \dots, N_k\}$  such that  $N = N'$ . That is, there is a top- $N$  query in  $P$  that is identical to the new query in both the query point and result size. In this case, let  $r := r'$ , where  $r'$  is from the profile  $\zeta' = (Q', N', r', f', c', d')$ .
- b. There is no  $N' \in \{N_1, N_2, \dots, N_k\}$  such that  $N = N'$  but there is  $N' \in \{N_1, N_2, \dots, N_k\}$  such that  $N' > N$  and it is the closest to  $N$  among  $N_1, N_2, \dots, N_k$  (Figure 1). In this case, let  $r := r'$  to guarantee the retrieval of all the top- $N$  tuples for  $Q$ .
- c.  $N_k < N$ . In this case, we assume that the search space for  $Q$  has the same local distribution density as that for  $Q'$ . Based on this assumption, we have  $N/(2r)^n = N_k/(2r_k)^n$ . As a result, we let  $r := (\sqrt[n]{N/N_k})r_k$ . If not enough top- $N$  tuples are retrieved, a larger  $r$  will be used (see Section 4.2.2).

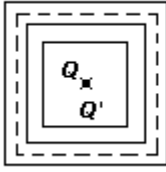


Fig. 1.  $Q = Q'$ .

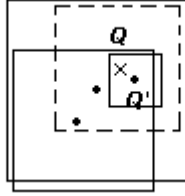


Fig. 2.  $Q \in S(Q', r', N')$ .

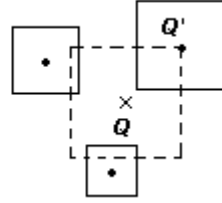


Fig. 3.  $Q \notin S(Q', r', N')$ .

2.  $d(Q, Q') \neq 0$ , i.e.,  $Q' \neq Q$ . We consider two subcases.

- a.  $Q$  is in the search space  $S(Q', r', N')$  of  $Q'$ . Find out all query points in  $P$  whose search spaces contain  $Q$ . Let these query points be  $(Q_1, \dots, Q_k)$  (see Figure 2). To estimate the search distance  $r$  for  $Q$ , we first use a weighted average of the local distribution densities of the search spaces for  $Q_1, \dots, Q_k$  to estimate the local density of search space for  $Q$ . The weight  $w_i$  for the search space corresponding to  $Q_i$  is computed based on its size and the distance between  $Q_i$  and  $Q$ . Weight  $w_i$  is an increasing function of the size of the search space and a decreasing function of the distance. In this paper,  $w_i$  is computed by the following formula:

$$w_i = v(S(Q_i, r_i, N_i)) * 1/(d(Q, Q_i))^\alpha$$

where  $\alpha$  is a parameter and  $\alpha = 3n/4$  is a good value based on our experiments. Let  $\rho_i = f_i/(2r_i)^n$  be the local density of the search space for  $Q_i$ . Then the local density of the search space for  $Q$  is estimated by:

$$\rho = (\sum_{i=1}^k w_i \rho_i) / (\sum_{i=1}^k w_i).$$

Based on the above  $\rho$ , we estimate the search distance  $r$  to be  $(\sqrt[n]{2N/\rho})/2$ .

Note that to increase the possibility that all of the top- $N$  tuples for  $Q$  are retrieved, we replaced  $N$  by  $2N$  in the estimation for  $r$  (i.e., aim to retrieve  $2N$  tuples).

- b.  $Q$  is not in the search space  $S(Q', r', N')$  of  $Q'$  (see Figure 3). Let  $h := d(Q, Q')$  be the distance between  $Q$  and  $Q'$ . Construct an  $n$ -square  $S(Q, h)$  and let  $(Q_1, \dots, Q_k)$  be all the query points in  $P$  whose search spaces intersect with  $S(Q, h)$ . Obviously,  $k \geq 1$  as  $Q'$  is in this query set. Now the same technique used above in step 2.a is used to estimate the search distance  $r$  for  $Q$ .

The search distance  $r$  obtained above (2.a and 2.b) may sometimes be either too small or too large. To remedy this, the following adjustments to  $r$  are implemented. The following two cases are considered.

(1)  $N = N'$ .

- (i) If  $r < r'$  or  $r < d(Q, Q')$ , then  $r$  may be too small. We use the following formula to adjust  $r$ :

$$r = \max(r\_Median, r\_Mean, r)/2 + (r' + d(Q, Q'))/2$$

where  $r\_Median = (\sqrt[3]{2N / N_{median}}) r_{median}$ ,  $r_{median}$  is the search distance of the search space whose density is the median among all search spaces in  $P$ , and  $N_{median}$  is the  $N$  value of the corresponding profile;  $r\_Mean = (\sqrt[3]{2N / \rho_{mean}})/2$  and  $\rho_{mean}$  is the average of all the densities of the search spaces in  $P$ .

- (ii) If  $r > r' + d(Q, Q')$ , then  $r$  is too large as  $r = r' + d_{\infty}(Q, Q')$  can already guarantee the retrieval of all the top- $N$  tuples of  $Q$ . In this case, we simply lower  $r$  to  $r' + d(Q, Q')$ .

(2)  $N \neq N'$ . This is handled in a similar manner as in case (1) except that a constant factor  $\lambda = \sqrt[3]{N / N'}$  is utilized to take into consideration the difference between  $N$  and  $N'$ .

- (i) If  $r < \lambda r'$  or  $r < d(Q, Q')$ , then  $r := \max(r\_Median, r\_Mean, r)/2 + (\lambda r' + d(Q, Q'))/2$ .
- (ii) If  $r > \lambda r' + d(Q, Q')$ , then  $r := \lambda r' + d(Q, Q')$ .

#### 4.2.2 Query Mapping Strategies

For a given top- $N$  query  $Q = (q_1, \dots, q_n)$ , to retrieve all tuples in  $S(Q, r)$ , one strategy is to map each top- $N$  query to a simple selection range query of the following format [1]:

**SELECT \* FROM R WHERE**  $(q_1 - r \leq A_1 \leq q_1 + r)$   
**AND ... AND**  $(q_n - r \leq A_n \leq q_n + r)$

If the query returns  $\geq N$  results, sort them in non-descending distance values and output the top  $N$  tuples. A potential problem that needs to be handled is that the estimated search distance  $r$  is not large enough. In this case, the value of  $r$  needs to be increased to guarantee that there are at least  $N$  tuples in  $S(Q, r)$ . One solution to this problem is provided below. Choose  $N$  query points in  $P$ , which are closest to the top- $N$  query  $Q$ , and sort them in ascending order of their distances with  $Q$  with respect to the used distance function  $d(.,.)$ . Let the order be  $Q_1, \dots, Q_N$  and their corresponding

profiles be  $\zeta_1, \zeta_2, \dots, \zeta_N$ . There exists a number  $h$ ,  $1 < h \leq N$ , such that  $N_1 + \dots + N_h \geq N$ . During the computation of the sum, if  $S(Q_i, r_i, N_i) \cap S(Q_j, r_j, N_j) \neq \emptyset$ , then  $N_i + N_j$  in the above sum was replaced by  $\max\{N_i, N_j\}$  to ensure that the search spaces of the first  $h$  queries in  $(Q_1, \dots, Q_N)$  contain at least  $N$  unique tuples. Let  $r := \max_{1 \leq i \leq h} \{d(Q, Q_i) + r_i\}$ . Thus, by using this  $r$  as the search distance for  $Q$  to generate the restart query, the restart query will guarantee the retrieval of all the top- $N$  tuples for  $Q$ .

If there is a *histogram* over the relation  $R$ , by using dNRq in [1], the search distance  $r$  for the restart query can be obtained as follows. If the sum of all  $N$ 's in  $P$  is less than  $N$ , then set  $r$  to be dNRq. Otherwise, find the number  $h$  as mentioned above and let  $r := \min\{\max_{1 \leq i \leq h} \{d(Q, Q_i) + r_i\}, \text{dNRq}\}$ .

## 5 Experimental Results

### 5.1 Data Sets and Preparations

All of our experiments are carried out using Microsoft's SQL Server 7.0 on a PC. To facilitate comparison, the same datasets and parameters used in [1] and [7] are used in this paper. These datasets include data of both low dimensionality (2, 3, and 4 dimensions) and high dimensionality (25, 50, and 104 dimensions). For low-dimensional datasets, both synthetic and real datasets used in [1] are used. The real datasets include Census2D and Census3D (both with 210,138 tuples), and Cover4D (581,010 tuples). The synthetic datasets are Gauss3D (500,000 tuples) and Array3D (507,701 tuples). In the names of all datasets, suffix KD indicates that the dataset has K dimensions. For high-dimensional datasets, real datasets derived from LSI are used in our experiments. They have 20,000 tuples and the same 25, 50 and 104 attributes as used in [7] are used to create datasets of 25, 50 and 104 dimensions, respectively. Each experiment uses 100 test queries (called a workload). The workloads follow two distinct query distributions, which are considered representatives of user behaviors [1]:

*Biased*: Each query is a random existing point in the dataset used.

*Uniform*: The queries are random points uniformly distributed in the dataset used.

The uniform workloads are only used for low-dimensional datasets to compare the results with those of [1]. For convenience, we report results based on a default setting. This default setting uses a 100-query Biased workload,  $N = 100$  (i.e., retrieve top 100 tuples for each query) and *max* is the distance function. When a different setting is used, it will be explicitly mentioned.

The most basic way of processing a top- $N$  query is the sequential scan method [4,5]. In this paper, we compare the following four top- $N$  query evaluation techniques:

*Optimum technique* [1]. As a baseline, we consider the ideal technique that uses the smallest search space containing the actual top- $N$  tuples for a given query. The smallest search space is obtained using the sequential scan technique in advance.



*Histogram-based techniques* [1]. We only cite the results produced by the dynamic (Dyn) mapping strategy described in [1] for comparison purpose. Dyn is the best among all histogram-based techniques studied in [1].

*Sampling-based technique* [7]. In this paper, we only cite the experimental results produced by the *parametric* (Para) strategy described in [7]. The Para strategy is the best of all sampling-based strategies discussed in [7].

*Learning-based (LB) technique*. This is our method described in Sections 4. For a given dataset  $D$ , the knowledge-base (or profile set)  $P$  is constructed as follows. First, a set of *random* tuples from  $D$  is selected. The size of the random tuple set is decided such that the size of  $P$  does not exceed the size of the histogram or the size of the sampling set when the corresponding method is being compared. For each query point chosen for  $P$ , the *sequential scan technique* is used to obtain its profile during the knowledge-base construction phase. For easy comparison, we use the following performance metrics used in [1] in this paper.

*Percentage of restarts*. This is the percentage of the queries in the workload for which the associated selection range query failed to retrieve the  $N$  best tuples, hence leading to the need of a restart query. When presenting experimental results,  $\text{Method}(x\%)$  is used to indicate that when evaluation strategy *Method* is used, the percentage of restart queries is  $x\%$ . For example,  $\text{LB}(3\%)$  means that there are 3% restart queries when the learning-based method is used.

*Percentage of tuples retrieved*. This is the average percentage of tuples retrieved from the respective datasets for all queries in the workload. When at least  $N$  tuples are retrieved, lower percentage of tuples retrieved is an indication of better efficiency. We report *SOQ* (*Successful Original Query*) percentages and *IOQ* (*Insufficient Original Query*) percentages. The former is the percentage of the tuples retrieved by the initial selection range query and the latter is the percentage of the tuples retrieved by a restart query when the initial query failed to retrieve enough tuples.

## 5.2 Performance Comparison

### 5.2.1 Comparison with Dyn

The experimental results that compared Dyn and Para were reported for low-dimensional datasets and for both *Biased* and *Uniform* workloads in [1] and the differences between the two methods are very small. Therefore, it is sufficient to compare LB with one of these two methods for low-dimensional datasets.

Figure 4 compares the performance of LB and that of Dyn for different datasets and for both *Biased* and *Uniform* workloads. From Figure 4(a), it can be seen that when *Biased* workloads are used, for datasets Gauss3D and Census3D, LB outperforms Dyn significantly; for Array3D, Census2D and Cover4D, LB and Dyn have similar performance. For *Uniform* workloads (Figure 4(b)), LB is significantly better than Dyn for 4 datasets and is slightly better for 1 dataset. However, LB has much higher restart percentages for Gauss3D and Cover4D (95% and 31%, respectively).

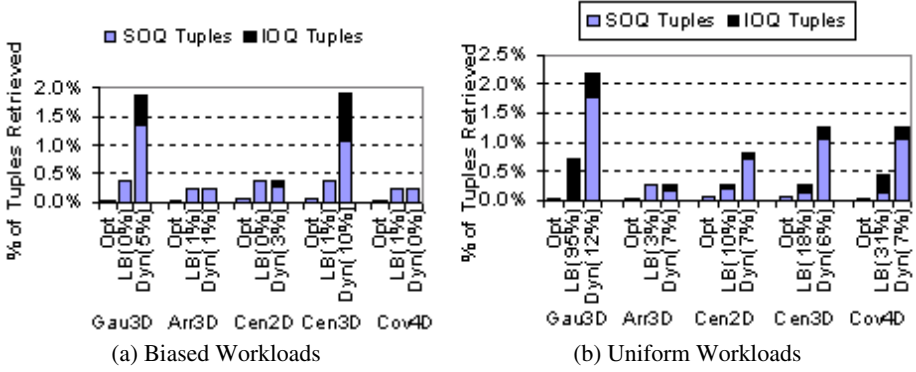


Fig. 4. Comparison of LB and Dyn.

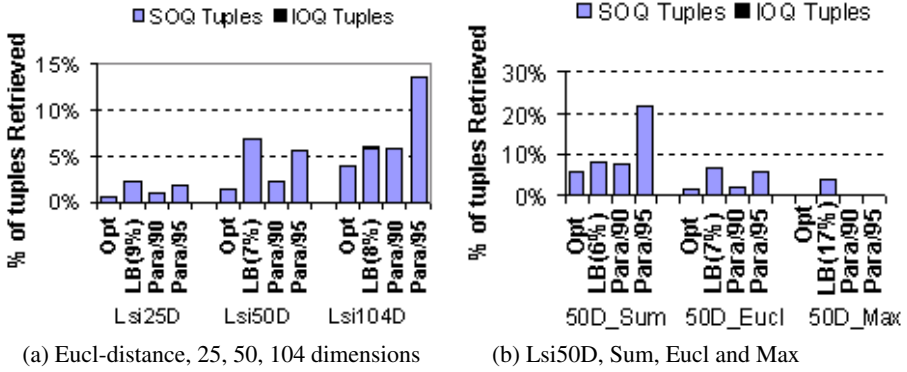


Fig. 5. Comparison of LB and Para.

### 5.2.2 Comparison with Para for High-Dimensional Datasets

Note that the Para method does not aim to guarantee the retrieval of all top- $N$  tuples. In [7], results for top-20 (i.e.,  $N=20$ ) queries when retrieving 90% (denoted Para/90) and 95% (denoted Para/95) were reported. In contrast, LB guarantees the retrieval of all top- $N$  tuples for each query. Figure 5 compares the performances of LB and Para for top-20 queries [7]. When Euclidean distance function is used, LB slightly underperforms Para/95 for 25- and 50-dimensional data but significantly outperforms Para/95 for 104-dimensional data. Figure 5(b) shows the results using different distance functions for 50-dimensional data. For the most expensive *sum* function, LB is significantly better than Para/95; for Euclidean distance function, Para/95 is slightly better than LB, and for the *max* function, Para/95 is significantly better than LB.

### 5.3 Additional Experimental Results for LB

We carried out a number of additional experiments to gain more insights regarding the behavior of the LB method. In this section, we report the results of these experiments.

**Effect of Different Result Size  $N$ .** Based on dataset Census2D, we construct 4 different knowledge bases using top-50, top-100, top-250 and top-1000 queries, respectively. For each knowledge base, the number of queries used is the same (1,459 queries). These knowledge bases are then used to evaluate a test workload of 100 top-100 queries. Our results indicate that using the knowledge bases of top-50 and top-100 queries yields almost the same performance. The best and the worst performances are obtained when the knowledge base of top-250 queries and that of top-1000 queries are used, respectively.

**Effect of Different Values of  $N$ .** In the second set of experiments, we use some top-100 queries to build a knowledge base for each of the 5 low-dimensional datasets. 178, 218 and 250 queries are used to build the knowledge base for 2-, 3- and 4-dimensional datasets, respectively. Each knowledge base is then used to evaluate a workload of top-50, top-100, top-250 and top-1000 queries. Overall, the results are quite good except when top-1000 queries are evaluated. But even for top-1000 queries, on the average, no more than 2% of the tuples are retrieved in the worst-case scenario. For this experiment, our results indicate that the LB method performs overall much better than the Dyn method. For example, in only two cases, more than 1% of the tuples are retrieved by LB, but in Figure 22(b) in [1], there are 10 cases where more than 1% of the tuples are retrieved.

**Effect of Different Distance Functions.** The effectiveness of the LB method may change depending on the distance function used. Our experimental results using various low-dimensional datasets indicate that the LB method produces similar performance for the three widely used distance functions (i.e., max, Euclidean and sum). By comparing our results with Figure 21(b) in [1], we find that the LB method significantly outperforms the Dyn method for datasets Gauss3D, Census3D and Cover4D; for the other two datasets, namely Array3D and Census2D, the two methods have similar performance.

## 6 Conclusions

In this paper, we proposed a learning-based strategy to translate top- $N$  selection queries into traditional selection queries. This technique is robust in several important aspects. First, it can cope with a variety of distance functions. Second, it does not suffer the much feared “dimensionality curse” as it remains effective for high-dimensional data. Third, it can automatically adapt to user query patterns so that frequently submitted queries can be processed most efficiently. We carried out extensive experiments using a variety of datasets of different dimensions (from 2 to 104). These results demonstrated that the learning-based method compares favorably with the current best processing techniques for top- $N$  selection queries.

## Acknowledgments

The authors would like to express their gratitude to Nicolas Bruno [1] and Chung-Min Chen [7] for providing us some of the test datasets used in this paper and for providing us some experimental results of their approaches that made it possible for us to compare with their results directly in this paper.

## References

1. N. Bruno, S. Chaudhuri, and L. Gravano. Top- $k$  Selection Queries over Relational Databases: Mapping Strategies and Performance Evaluation, *ACM Transactions on Database Systems*, 27 (2), 2002, 153–187.
2. N. Bruno, S. Chaudhuri and L. Gravano. STHoles: A Multidimensional Workload-Aware Histogram. ACM SIGMOD Conference, 2001.
3. N. Bruno, L. Gravano, and A. Marian. Evaluating Top- $k$  Queries over Web-Accessible Databases. 18th IEEE International Conference on Data Engineering, 2002.
4. M. Carey, and D. Kossmann. On saying “Enough Already!” in SQL. ACM SIGMOD Conference, 1997.
5. M. Carey, and D. Kossmann. Reducing the braking distance of an SQL query engine. International Conference on Very Large Data Bases. 1998.
6. S. Chaudhuri, and L. Gravano. Evaluating top- $k$  selection queries. International Conference on Very Large Data Bases, 1999.
7. C. Chen, and Y. Ling, A sampling-based estimator for top- $k$  selection query. 18th International Conference on Data Engineering, 2002.
8. C. Chen and N. Roussopoulos. Adaptive selectivity estimation using query feedback. ACM SIGMOD Conference, 1994, 161–172.
9. Y. Chen, and W. Meng. Top-N Query: Query Language, Distance Function and Processing Strategies. International Conference on Web-Age Information Management, 2003.
10. D. Donjerkovic, and R. Ramakrishnan. Probabilistic optimization of top N queries. International Conference on Very Large Data Bases, 1999.
11. R. Fagin. Combining fuzzy information from multiple systems. ACM Symposium on Principles of Database Systems, 1996, 216-226.
12. R. Fagin. Fuzzy queries in multimedia database systems. ACM Symposium on Principles of Database Systems, 1998.
13. R. Fagin, A. Lotem, and M. Naor. Optimal aggregation algorithms for middleware. ACM Symposium on Principles of Database Systems, 2001.
14. J. Lee, D. Kim, and C. Chung. Multi-dimensional Selectivity Estimation Using Compressed Histogram Information. ACM SIGMOD Conference, 1999, 205-214.
15. C. Yu, G. Philip, and W. Meng. Distributed Top-N Query Processing with Possibly Uncooperative Local Systems. VLDB, Berlin, Germany, September 2003, 117-128.

# DHT Based Searching Improved by Sliding Window

Shen Huang, Gui-Rong Xue, Xing Zhu, Yan-Feng Ge, and Yong Yu

Department of Computer Science and Engineering, Shanghai Jiao Tong University,  
Huashan Avenue 1954, 200030 Shanghai, P.R.China  
{Huangshen, grxue, ZhuXing, gyf}@sjtu.edu.cn  
yyu@cs.sjtu.edu.cn

**Abstract.** Efficient full-text searching is a big challenge in Peer-to-Peer (P2P) system. Recently, Distributed Hash Table (DHT) becomes one of the reliable communication schemes for P2P. Some research efforts perform keyword searching and result intersection on DHT substrate. Two or more search requests must be issued for multi-keyword query. This article proposes a Sliding Window improved Multi-keyword Searching method (SWMS) to index and search full-text for short queries on DHT. The main assumptions behind SWMS are: (1) query overhead to do standard inverted list intersection is prohibitive in a distributed P2P system; (2) most of the documents relevant to a multi-keyword query have those keywords appearing near each other. The experimental results demonstrate that our method guarantees the search quality while reduce the cost of communication.

## 1 Introduction

Peer-to-peer (P2P) system becomes popular in recent years. In such system, DHT [5, 16, 19, 24] efficiently performs object location and application routing in a potentially very large overlay network. Normally, the object to be placed or located has an ID. Meanwhile the node where the object is placed also has an ID taken from the same space as the object's key. When a node receives a query for a key for which it is not responsible, the node routes the query to the neighbor node that makes the most "progress" towards resolving the query. Each node maintains a routing table consisting of a small subset of nodes in the system and ensure a relative short route path.

One main problem for P2P is how to retrieve global information on the distributed infrastructure. Napster [15] and Gnutella [26] provide good strategies for title-based file retrieval. However, indexing full-text in such environment hasn't been solved well enough. Some researches [3, 17, 25] performed keyword searching on DHT substrate. All of them adopted storing (term, index) pairs for each term appearing in each document on corresponding node. Such indexing method is a kind of global index defined in [2]. Using global index, multi-keyword searching will generate several requests and intersect the result sets. Ribeiro-Neto [4] showed that only in a tightly coupled network which has high bandwidth connection, such index will work well. However, such connection is always unavailable in today's P2P network.

To reduce the communication costs, one alternative is indexing the combination of any two or more terms in a document [10]. After that, multi-keyword searching needs to access only one node. Normally such method is unfeasible because of the huge amount of term combinations. Here we make a tradeoff between communication and index costs and show that the performance is acceptable. Li [13] mentioned that one important ranking factor is how close the keywords occur to each other. Proximity search [9] is used in information retrieval to locate documents that have words occurring “near” each other. Moreover, users are very likely to submit smaller queries with 2 or 3 keywords [17, 21]. Based on the proximity assumption and users’ behaviors, we use sliding window to handle 2-keyword or 3-keyword searching. By searching combination index instead of several individual index, we achieve a good approximation result for short queries with lower communication costs.

The rest of the paper is organized as follows. Some related work are discussed in Section 2. Section 3 introduces some definitions and three search mechanisms. Section 4 describes how to efficiently build bi/tri-term index based on proximity approximation and vocabulary reduction. The experimental results of our method are presented in Section 5. Finally, Section 6 gives the conclusion and the directions of future work.

## 2 Related Work

Some researches [3, 17, 25] aim to perform keyword searching on the DHT-enabled substrate. Tang [25] built a scalable P2P IR system that has efficiency of DHT systems and accuracy of state-of-the art IR algorithm upon a hierarchical version of CAN. Reynolds [17] focused on minimizing network traffic for multi-keyword queries. One technique is Bloom filter [11], a hash-based data structure that summarizes membership in a set. Bhattacharjee [3] proposed another technique to improve efficiency: result caching. This orthogonal and complementary technique avoids duplicating work and data movement. Different with these methods, SWMS reduces communication cost using combined index. It requires more storage and computing time. With appropriate pruning, such overhead can be reduced to an acceptable level.

Gnawali [10] proposed using window to build index for “term set”. The idea is similar to SWMS, but it doesn’t explore how window-based method will affect the retrieval result. Clarke [7] and Rasolofo [28] both used proximity assumption to improve retrieval performance. We make similar assumption, but the goal is different: we try to limit the size of huge combined index. Document Frequency and Mutual Information method are used by [27] to filter out non-important features in text classification. SWMS borrows the ideas and make the index more compact.

Tang [25] and Reynolds [17] proposed to rank documents in the intersection of result set. But intersection of single keyword results may be not good for ranking multi-keyword result. Moreover, compression techniques such as Bloom filters [11] complicate problem of ranking results and some trade-offs have been determined [17]. SWMS differs in that it considers several keywords simultaneously and ranks the result using their positions and frequencies in a document.

### 3 Multi-keyword Search Mechanism

Before introducing search mechanism, we first define *bi-term* and *tri-term*. We analyze the relevance of words by counting their co-occurrences in a span of text.

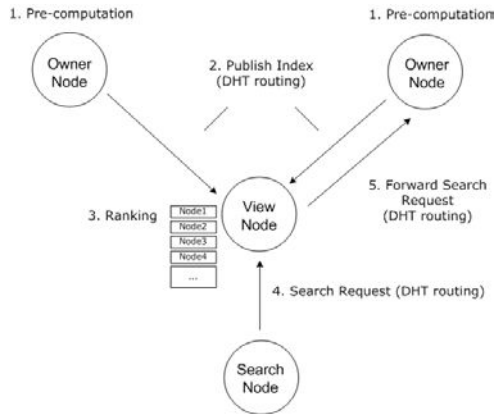
$$\begin{cases} Bi(t_1, t_2) \equiv t_1 \wedge_s t_2 \\ Tri(t_1, t_2, t_3) \equiv t_1 \wedge_s t_2 \wedge_s t_3 \end{cases} . \quad (1)$$

where  $t_1$ ,  $t_2$  and  $t_3$  are three terms.  $\wedge_s$  means two terms occur in the same span of text. Here we don't consider the sequence of the terms.  $Tri(t_1, t_2, t_3)$  and  $Bi(t_1, t_2)$  are tri-term and bi-term respectively. In the rest of this paper, they are both called *combined term* and the index of them are called *combined index*.

#### 3.1 Search Mechanism

In this section we'll introduce our search mechanism for multi-keyword. As Figure 1 shows, owner node has the documents the search nodes want to retrieve. And the view node stores the combined index and rank documents. We use DHT scheme [5, 16, 19, 24] to publish and retrieve such information. By doing this, our approach can perform searching in two steps. First, with the keywords submitted by a user, search node generates a view and directly forward request to corresponding view node using DHT scheme. Second, with the published index from different owners, view node chooses some highly ranked results and return them to search node.

To compare our work with others, we introduce three search mechanisms as Figure 2 shows. Mechanism 1 is used by SWMS to perform search with combined index. To intersection method, two choices exist. "Load first" means the results are intersected as soon as possible to reduce the load of whole network. We defined it as mechanism 2. "Time first" means results are returned to users as soon as possible, which is defined as mechanism 3. Figure 2 shows how they work for three-keyword query. Obviously, mechanism 1 reduces communication cost and query latency.



**Fig. 1.** Basic search mechanism. Each node can have three roles during the search.

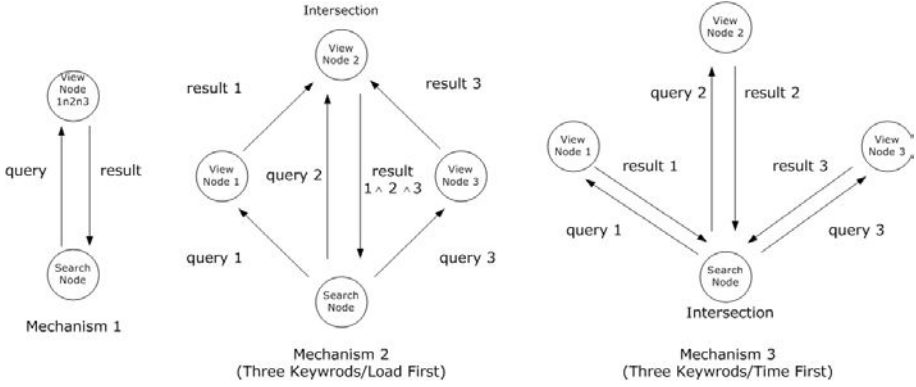


Fig. 2. Three mechanisms for multi-keyword searching. Mechanism 1 search more efficiently.

## 4 Efficient Pre-computation

If we want to leverage the location and routing capacities of DHT substrate using search mechanism 1, we must index the combination of several terms. However, computing the occurrences of combined terms is highly expensive. For a 300-word document, the combined terms will exceed 4 million! Thus we borrow the ideas from proximity search [9] and define the boundary for the term to be indexed. The span between two boundaries is defined as a window. When window slides over the whole document, only the terms within the same window will be combined. Based on such sliding window, we revise the document ranking by assume that:

- The closer a group of term occurrences in a window, the more likely that the corresponding window is relevant
- The more relevant windows contained in a document, the more likely that the document is relevant

The first assumption means that we should give more weight to the more “near” co-occurrence. The second one means that the more co-occurrences, the higher rank the document has. The ranking functions for combined term are listed as follows:

$$\begin{cases}
 Weight(Bi(t_1, t_2)) = \log \sqrt{\frac{|SW|}{|q-p|}} \\
 Rank(Bi(t_1, t_2)) = \sum_{i=0}^N Weight(Bi(t_1, t_2))
 \end{cases}
 \quad .$$

$$\begin{cases}
 Weight(Tri(t_1, t_2, t_3)) = \log \sqrt{\frac{|SW|}{|q-p|} + \frac{|SW|}{|p-o|} + \frac{|SW|}{|q-o|}} \\
 Rank(Tri(t_1, t_2, t_3)) = \sum_{i=0}^N Weight(Tri(t_1, t_2, t_3))
 \end{cases}
 \quad .$$
(2)

here  $|SW|$  is length of sliding window,  $o$ ,  $p$  and  $q$  are the positions of term  $t_1$ ,  $t_2$  and  $t_3$  in document.  $N$  is number of co-occurrences of a bi-term/tri-term in a document.



#### 4.1 Indexing Pruning

Sliding window ensures the co-occurred terms near to each and largely reduces the indexing size. Nevertheless, the indexing amount is still too large to store and retrieve. On one of our test set TIME [22], a 5-term window (i.e. the length of the window is 5) will generate indexing items over 950,000, while single terms are no more than 24,000. We use two methods introduced in [27] to filter out unimportant items for query.

The first one is Document Frequency Thresholding (DF). Document frequency is the number of documents in which a term occurs. This is the simplest technique for vocabulary reduction. It's easily scales to very large corpora with a computational complexity approximately linear in the number of documents. Notice that much of the bi-term or tri-term indexing items occur in only one document, we think such co-occurrences are occasional and can be ignored.

The second method is Mutual Information (MI). DF is typically not used for aggressive term removal because of a widely received assumption in information retrieval. That is low-DF terms are assumed to be relatively informative and therefore should not be removed aggressively. MI is a criterion commonly used in statistical language modeling of word associations [12]. Consider  $A$  is the number of times a term  $t1$  and  $t2$  co-occur,  $B$  is the number of times  $t1$  occurs without  $t2$ ,  $C$  is the number of times  $t2$  occurs without  $t1$ , and  $N$  is the total number of documents, then the MI between  $t1$  and  $t2$  is defined to be

$$I(t_1, t_2) = \log \frac{Pr(t_1 \wedge t_2)}{Pr(t_1) \times Pr(t_2)} \quad . \quad (3)$$

and is estimated using

$$I(t_1, t_2) \approx \log \frac{A \times N}{(A + C) \times (A + B)} \quad . \quad (4)$$

MI will enable rare terms have a higher score than common terms. Using both DF and MI methods, we try to reduce nonsense indexing items while reserve rare but informative items.

In this section, we describe how to largely reduce the indexing cost for combination operation. Next, we'll show the performance of this tradeoff is acceptable on SMART IR test sets [6, 22].

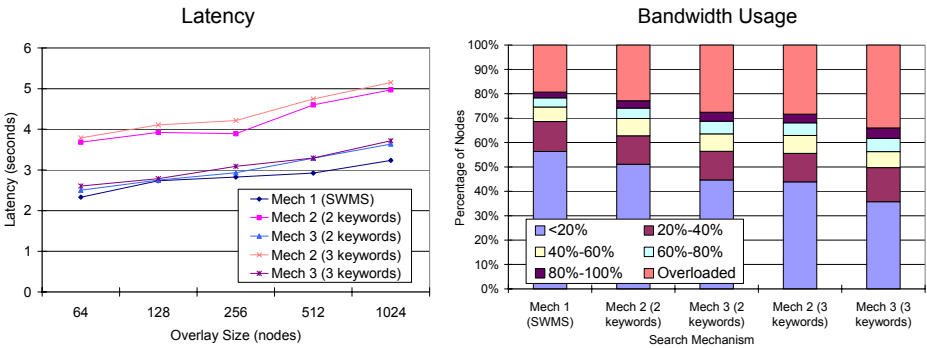
## 5 Experiment

SWMS makes a tradeoff between communication and indexing cost. Besides, the retrieval performance will be considered. The experiment is divided into three main parts: search performance, indexing overhead and retrieval quality. To establish a simulated network, we first use Brite [1] to create a network topology, on which DHT substrate is built using Pastry [18]. Next, we randomly select about 200 files from a collection [22] for each node. Finally, SWMS is implemented in following steps:

- For each document, sliding window is used to count the occurrences of combined term as well as single term. The sequence of terms constituting a combined term is ignored. We sort and append them one by one using “\_” to ensure a unique representation. Pruning is used to get a more compact indexing.
- Each node hashes both the combined term and single term use secure hash [8]. Then publish the index to corresponding nodes using DHT scheme.
- The view node ranks documents by the indexing received from other nodes, which search node uses to locate view node.

### 5.1 Query Latency and Network Load

To get more practical result, we use the data from [23]. In that experiment, authors measure latency to a total of 339,502 Gnutella [26] peers, upstream bandwidths from 16,252 peers and downstream from 223,552 peers. According to the measured data, we assign the latency and bandwidth between each two peers. Every request is firstly routed by Pastry substrate. Then the request is routed in topology and the involved links are marked with some band consumption and latency. The file transferring from owner node to search node is ignored because it's same for SWMS and intersection method. We create 100,000 three-keyword searching requests using the queries in [22] to get latency measurement on the left of Figure 3. To network load, we perform 1000 queries in overlay size of 1024 nodes and get the right of Figure 3.



**Fig. 3.** Left is latency for different mechanisms. Latency of SWMS increases more slowly with the size of overlay. Right is network load. We make some assumptions about the bandwidth consumption. Reality is more complex, but this is out of the scope of the paper. What we want to show is the intersection method does consume more bandwidth.

### 5.2 Indexing Overhead

After observing the performance of search mechanisms, let's have a close look at indexing overhead. We use six collections: TIME, MED, NPL, ADI, CACM and CRAN [22]. We first filter out stop word, then stem the rest using Porter Stemming algorithm [14]. Some characteristics of the test sets are listed:

**Table 1.** The test collections. “Single indexing term” means the non-combined term.

Test Set	Doc	Queries	Single Indexing Terms	Average Terms/Query	Average Words/Doc
TIME	425	83	24,000	9.05	234.66
MED	1033	30	14,585	10.27	61.08
NPL	11429	93	26,815	10.86	19.74
ADI	82	35	970	14.26	28.49
CACM	3212	64	10,002	22.28	24.60

We use a PC with 667MHZ CPU and 256M memory to build inverted index. Windows lengths of 5, 25 and 45 words are tried on each collection. Moreover, we want to know whether the co-occurrence of the words within one sentence is important. Sentence windows don’t overlap each other. The indexing is pruned using Document Frequency mentioned in Section 4. Mutual Information is not used here and we’ll discuss it later. DF threshold is set to 2, i.e. any index whose document frequency less than 2 will be pruned. Here we take TIME collection for example. Table 2 lists some statistics. We notice that the pre-computation does boost the index size even if sliding window is used. But the DF-based pruning cuts most of the combined items. The overhead of index storage and publish can be restricted to an acceptable level. Next, we’ll explore whether pruning degrade retrieval performance.

**Table 2.** TIME indexing overhead, DF threshold is 2. “No Win” means intersection method not using sliding window. The number in the brackets is the index size before pruning.

	SW = 5	SW = 25	SW = 45	SW = 1 sentence	No Win
Number of Indexing Terms (No pruning)	46,626 ( $9.6 \times 10^5$ )	87,023 ( $3.8 \times 10^6$ )	88,979 ( $3.9 \times 10^6$ )	78,265 ( $3.5 \times 10^6$ )	$2.4 \times 10^4$
File Size (KB)	3,319	6,068	6,096	5,956	2,057
Indexing Time/Doc (ms)	32	167	255	214	6

### 5.3 Retrieval Quality

We use precision and recall [20] to explore retrieval performance of SWMS. Because only short queries with three or less keywords are handled, we invite a group of students to shorten the collections’ long queries. Everyone chooses at most 5 keywords he/she think important for each query and the most popular 3 keywords will be adopted. Although such shorten queries may degrade the performance of retrieval, we find that SWMS still work well. The results are ranked using the method mentioned in Section 4. The quality is very close to that of intersection method. Table 3 shows average “Top 10” performance for the 6 collections:

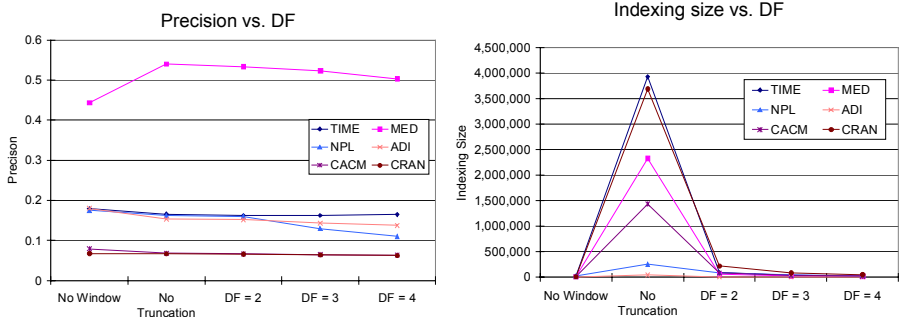
The window size has little effect on the retrieval result. To each collection, the difference is no more than 4 percent. And the ideal window size has no explicit relation with document length. TIME get the best result using the smallest window although it has long documents. Three out of six collections get the best result when the window is one sentence. So the assumption that the words in one sentence are more related to

**Table 3.** Top 10 precision and recall for all collections, DF threshold is 2. For each collection, the italic numbers are the best results among different sliding windows. The number in the brackets is the ratio of SWMS and Intersection performance (SWMS / No Win).

Test Sets	Top 10	SW = 5	SW = 25	SW = 45	SW = one sentence	No Win
TIME	Pre	<i>0.17(94%)</i>	0.16	0.16	0.16	0.18
	Rec	<i>0.46(87%)</i>	0.46	0.46	0.46	0.53
MED	Pre	0.52	0.52	<i>0.53(120%)</i>	<i>0.53(120%)</i>	0.44
	Rec	0.24	0.24	<i>0.24(120%)</i>	<i>0.24(120%)</i>	0.20
NPL	Pre	0.16	0.15	<i>0.17 (94%)</i>	0.16	0.18
	Rec	0.75	0.73	<i>0.77 (97%)</i>	0.75	0.79
ADI	Pre	0.15	<i>0.15 (83%)</i>	<i>0.15 (83%)</i>	<i>0.15 (83%)</i>	0.18
	Rec	0.35	<i>0.36 (95%)</i>	<i>0.36 (95%)</i>	<i>0.36 (95%)</i>	0.38
CACM	Pre	0.05	0.07	0.07	<i>0.07 (88%)</i>	0.08
	Rec	0.14	0.17	0.17	<i>0.18 (95%)</i>	0.19
CRAN	Pre	0.06	<i>0.07(100%)</i>	0.07	0.07	0.07
	Rec	0.17	<i>0.20(105%)</i>	0.19	0.19	0.19

each other may be reasonable. MED and CRAM get better result when using sliding window. To others, the best approximate results are also very close to the no window method. So sliding window can be used to limit combined index size without obvious degrade of retrieval performance.

For we use DF method to prune the combined index, we wonder how it affects the retrieval quality. We set DF threshold as 2, 3 and 4 and compare the results with no pruning method, as the left of Figure 4 shows. When DF is 2, the lost of precision is the least, less than 1 percent. When DF is 3 or 4, the lost become obvious. Unexpectedly, when we set DF as 4 for TIME, the precision improve slightly. This means sometimes high frequency common words are more important than low frequency words in retrieval. The right of Figure 4 shows the indexing size pruned by different



**Fig. 4.** Left is the precision using different DF. Right is indexing terms using different DF.

DF thresholds. We can see that when DF is 2, the pruning is the most cost effective. So we set DF threshold as 2 in above experiments.

We also want to reserve some rare but informative combined term using MI method (see Section 4.2). MI can be used to indicate the connection strength between two co-occurred words. Normally, when MI is more than zero the two words are related. In Table 4, “DF = 2, MI = 0” means when we do DF2 pruning, we reserve the combined terms whose MI is larger than zero. We can see the improvement of average “Top 10” search is slight when window size is 45. So we don’t adopt MI to improve search result in above experiments.

**Table 4.** Top 10 precision and recall for all collections, the size of sliding window is 45.

	Top 10	TIME	MED	NPL	ADI	CACM	CRAN
DF = 2	Pre	0.16 (94%)	0.53 (98%)	0.17 (100%)	0.15 (94%)	0.07 (100%)	0.07 (100%)
	Rec	0.46 (96%)	0.24 (96%)	0.77 (99%)	0.36 (97%)	0.17 (94%)	0.19 (95%)
DF = 2 MI = 0	Pre	0.16 (94%)	0.54 (100%)	0.17 (100%)	0.15 (94%)	0.07 (100%)	0.07 (100%)
	Rec	0.47 (98%)	0.25 (100%)	0.78 (100%)	0.37 (100%)	0.17 (94%)	0.20 (100%)
No Pruning	Pre	0.17	0.54	0.17	0.16	0.07	0.07
	Rec	0.48	0.25	0.78	0.37	0.18	0.20

## 6 Conclusion and Future Work

SWMS leverages sliding window and DHT routing to perform searching in shorter time without degrade of retrieval quality. Our work has four contributions:

- Proposing and compare three searching mechanisms.
- Using pruning to make combined indexing more practical.
- Introducing ranking function to ensure the quality of multi-keyword searching.
- Giving experimental evaluation on searching and ranking performance.

In future work, we should make our approach more strong to the dynamic P2P network. Moreover, current test collections’ queries are too long for SWMS. We plan to perform further experiment using some Web users’ logs. Finally, we try to apply sliding window method to the widely used VSM model [20]. In distributed environment, IDF is hard to estimate and more problems need to be solved.

## References

1. A. Medina, A. Lakhina, I. Matta, and J. Byers.: BRITE: An Approach to Universal Topology Generation, In Proc. of MASCOTS, 2001.

2. A. Tomasic and Hector Garcia-Molina.: Performance of Inverted Indices in Shared-nothing Distributed Text Document Information Retrieval Systems. In Proc. of PDIS '93, 1993
3. B. Bhattacharjee, S. Chawathe, V. Gopalakrishnan, P. Keleher, and B. Silaghi.: Efficient Peer to Peer Searches Using Result-Caching. In Proc. of IPTPS'03, 2003.
4. Berthier A. Ribeiro-Neto and Ramurti A. Barbosa.: Query Performance for Tightly Coupled Distributed Digital Libraries. In Proc. of ACM DL'98, 1998.
5. Ben Y. Zhao, J. Kubiawicz, Anthony D. Joseph.: Tapestry: An Infrastructure for Fault-tolerant Wide-area Location and Routing. Tech, Rep, 2001.
6. C.Buckley: Implementation of the SMART Information Retrieval System. Tech. Rep. 1985
7. Charles L. A. Clarke, Gordon V. Cormack, Forbes J. Burkowski.: Shortest Substring Ranking (MultiText Experiments for TREC-4). NIST special publication.
8. FIPS 180-1. Secure hash standard. Tech. Rep. Publication 180-1, FIPS, 1995.
9. G. Salton.: Automatic Text Processing: The transformation, analysis, and retrieval of information by computer. Addison-Wesley, 1989.
10. O. D. Gnawali.: A Keyword-Set Search System for Peer-to-Peer Networks. Master's Thesis, Massachusetts Institute of Technology, 2002.
11. Burton H. Burton.: Bloom: Space/time Trade offs in Hash Coding with Allowable Errors. Communications of the ACM, 1970.
12. K. W. Church and P. Hanks.: Word Association norms, mutual information and lexicography. In Proc. of ACL 27, 1989.
13. J. Li, B. T. Loo, J. M. Hellerstein, M. F. Kaashoek, D. R. Karger, R. Morris.: On the Feasibility of Peer-to-Peer Web Indexing and Search. In Proc. of IPTPS '03, 2003.
14. M. F. Porter. An algorithm for suffix stripping. Program, 1980.
15. Napster: <http://www.napster.com>
16. A. Rowstron and P. Druschel.: Pastry: Scalable, Decentralized Object Location and Routing for Large-scale Peer-to-peer Systems. In Proc. of Middleware'01, 2001.
17. P. Reynolds and A. Vahdat.: Efficient Peer-to-Peer Keyword Searching. In Proc. of Middleware'03, 2003.
18. Pastry Software. <http://research.microsoft.com/~antr/Pastry/>
19. S. Ratnasamy, P. Francis, M. Handley, R. Karp, R., and S. Shenker.: A Scalable Content-Addressable Network. In Proc. of the ACM SIGCOMM'01, 2001
20. R. B. Yates, B. R. Neto.: Modern Information Retrieval. Addison-Wesley Pub Co, 1999
21. C. Silverstein, M. Henzinger, H. Marais, and M. Moricz.: Analysis of a Very Large Web Search Engine Query Log. In Proc. of ACM SIGIR, 1999.
22. SMART Test Collections: <http://www.cs.utk.edu/~lsi/corpa.html>
23. Stefan Saroiu, P. Krishna Gummadi, Steven D. Gribble.: A Measurement Study of Peer-to-Peer File Sharing Systems, In the Proc. of MMCN'02, 2002.
24. I. Stoic, R. Morris, D. Kargeer, M. F. Kaashoek, H. Balakrishnan.: Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications. In Proc of the SIGCOMM'01, 2001.
25. C. Q. Tang, Z. C. Xu, and S. Dwarkadas.: Peer-to-Peer Information Retrieval Using Self-Organizing Semantic Overlay Networks. In Proc. of the ACM SIGCOMM'03, 2003.
26. The Gnutella Homepage: <http://gnutella.wego.com>
27. Yiming Yang and Jan O. Pedersen.: A Comparative Study on Feature Selection in Text Categorization. In Proc. of ICML'97, 1997.
28. Y. Rasolofo and J. Savoy.: Term proximity scoring for keyword-based retrieval systems. In Proc. of ECIR'03, 2003.

# A Scalable and I/O Optimal Skyline Processing Algorithm

Yi Luo, Hai-Xin Lu, and Xuemin Lin

School of Computer Science & Engineering  
University of New South Wales  
Sydney, NSW 2052, Australia

{luoyi, cshlu, lxue}@cse.unsw.edu.au

**Abstract.** In this paper, we will present a novel progressive algorithm for computing skyline queries. The algorithm is based on the depth-first search paradigm. It can be shown that the new algorithm is I/O optimal. Further, we can show that the algorithm takes a logarithmic memory space in  $2D$ -space in the worst case if there are not many intersections in an adopted  $R$ -tree. Our experiment demonstrated that the new algorithm is more scalable and efficient than the existing techniques especially in a low dimensional space. The experiment also showed that the algorithm is progressive in a way sensitive to a user's preference.

## 1 Introduction

Given a set of points, “skyline query” returns the points in the set, which are not “dominated” by another point (See Figure 1 for example). Skyline computation is fundamental to many applications, including multi-criteria decision making [14] where the optimal criteria are some times conflicting. Consider the query “finding a cheap car with a recent model”. This is a typical example involving the two conflicting goals, cheap price and recent model. Instead of identifying an optimal answer, it may only be possible to provide a set of candidate answers that likely meet the user requirements. Clearly, some users may be able to afford a car with the latest model, while others may only be able to choose cars as new as possible within their budgets. In this application, it is unlikely that one will consider cars both older and more expensive than another; thus, the set of candidate cars form skyline.

The study of skyline query processing can be traced back to the 70's [6, 1]. Very recently, three  $R$ -tree based skyline processing techniques [7, 12, 9] have been proposed. The algorithm (NN) [7] is the first algorithm for computing skyline based on  $R$ -trees [2, 4], which uses the nearest neighbour search techniques [5, 13]. Observe [12] the heavy I/O overheads in NN especially in a high dimensional space. The algorithm (BBS) [12] was developed to minimise the I/O costs, which extended the best-first search technique [5] for the nearest neighbour problem. As reported in [12], BBS guarantees the minimum I/O overheads but requires a larger memory space than that in NN in the two dimensional space. In fact, we will show in section 2 that the algorithm BBS may take  $O(N)$  memory space where  $N$  is the number of data points. Clearly, less the memory space requires, more scalable the algorithm is. Motivated by this, our previous work, the

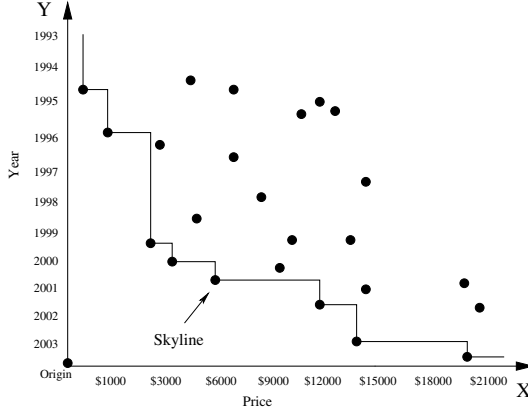


Fig. 1. Skyline Example

algorithm DCSkyline [9], based on a divide-conquer paradigm for computing skyline on a  $2D$  space has been developed to reduce space complexity. However, the progressive direction of the algorithm DCSkyline is independent on user's preference; and thus, it cannot be used as a progressive processing algorithm to accommodate user's requests.

In this paper, we will present a novel  $R$ -tree based algorithm, which adopts a depth-first search technique. In the algorithm, we also developed a “forward checking” technique based on a “region dominance” relation to reduce space complexity. We can show that the new algorithm is I/O optimal and requires a logarithmic space in the worst case in the  $2D$  space if there are not many overlapping in  $R$ -tree. Further, our experiment demonstrated that the new algorithm requires a much smaller space than the existing I/O optimal techniques, and is progressive in a way sensitive to user's requirements.

## 2 Preliminary

This section provides a brief overview of the  $R$ -tree based skyline query algorithms BBS and DCSkyline.

### 2.1 Definitions

Given a set of points  $S$  in the  $d$  dimensional space, a point  $p$  in  $S$  is a *skyline point* of  $S$  if  $p$  is not *dominated* by another point in  $S$ . A point  $(x'_1, x'_2, \dots, x'_d)$  *dominates* another point  $(x_1, x_2, \dots, x_d)$  if  $x'_i \leq x_i (i = 1, 2, \dots, d)$ . The *skyline* of  $S$  is the set of skyline points of  $S$ . An *entry* in  $R$ -tree refers to a node's MBB (Minimal Bounding Box) or a data point. As depicted in Figure 2, each  $e_i$  (for  $0 \leq i \leq 7$ ) is an entry, as well as the data points from  $a$  to  $n$ .

In an  $R$ -tree on a  $d$ -dimensional space, the *lower-left corner* of an intermediate entry  $[x_1^l, x_1^h] \times [x_2^l, x_2^h] \times \dots \times [x_d^l, x_d^h]$  is the point  $(x_1^l, x_2^l, \dots, x_d^l)$ , while the *upper-right corner* of the entry is the point  $(x_1^h, x_2^h, \dots, x_d^h)$ . An intermediate entry is dominated by a point  $p$  if the lower-left corner of the entry is dominated by  $p$ .



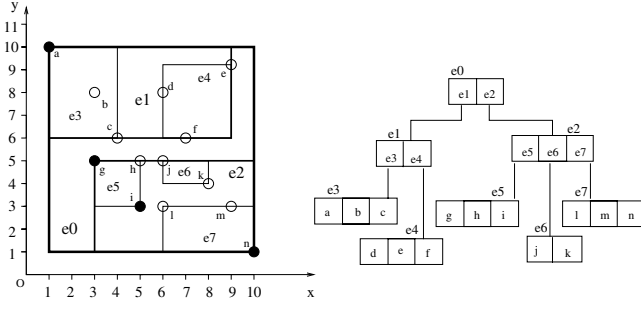


Fig. 2. R-Tree Example

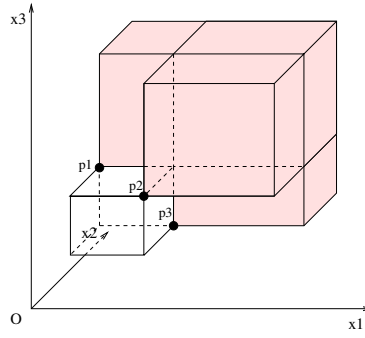


Fig. 3. Region Dominance

The dominance region ( $DR$ ) of an intermediate entry on a  $d$ -dimensional space is formalised as follows. In Figure 3, consider an entry  $e$  in a  $d$ -dimensional space whose MBB has ranges  $[x_1^l, x_1^h] \times [x_2^l, x_2^h] \times \dots \times [x_d^l, x_d^h]$ . Define  $d$  points  $p_1(x_1^l, x_2^h, \dots, x_d^h)$ ,  $p_2(x_1^h, x_2^l, \dots, x_d^h)$ , ...,  $p_d(x_1^h, x_2^h, \dots, x_d^l)$ . Then  $DR(e) = DR(p_1) \cup DR(p_2) \cup \dots \cup DR(p_d)$ .

Now we explain this formula. Since the entry  $e$  is a minimal bounding box, there must exist at least one point in  $e$ , whose coordinate on the first axis is  $x_1^l$ . Denote the point as  $q_1(x_1^l, x_2, \dots, x_d)$ . Considering another point  $p_1(x_1^l, x_2^h, \dots, x_d^h)$ , it is obvious that  $q_1$  dominates  $p_1$ . Thus, if an  $R$ -tree entry is dominated by  $p_1$ , it must also be dominated by  $q_1$ , and can be eliminated immediately. With this observation,  $p_1$  can be used as a pruning seed. The points  $p_2, p_3, \dots, p_d$  are defined similarly. Thus for the entry  $e$ , the region dominance checking can be done by the dominance checking with  $p_1, p_2, \dots$  and  $p_d$ . An entry is dominated by  $e$  if it is dominated by any of these  $d$  points.

## 2.2 Branch-and-Bound Skyline (BBS)

BBS [12] applies  $R$ -trees based nearest neighbour search techniques [5, 13], Nearest neighbour search iteratively returns the nearest points to a given query region. The algorithm needs to define a *distance* from an entry to a point  $p$ ; the minimum distance

between  $p$  and a point in the entry is used for such purpose. In this paper, we denote the distance from an entry to the query point by  $minDist$ .

The algorithm BBS developed by Papadias et al. extended the best-first search technique [5] by keeping all the candidate entries in a *heap* until they are no longer useful. Elements (entries) in the heap are based on their  $minDist$  values (to the origin), and then the skyline points are generated iteratively as the nearest neighbour points to the origin. For the dominance validation, the skyline points generated already are kept in a list  $S$  in the main memory.

Initially, the root of the  $R$ -tree is stored in the heap. Every time the first element  $e$ , which has the minimal  $minDist$  is compared with the skyline list  $S$ . The dominated  $e$  is discarded. Then the algorithm checks if  $e$  is a leaf node of the  $R$ -tree. In the case that  $e$  is a leaf, it is output as a skyline point, and inserted into the skyline list  $S$ . Otherwise, the children entries of  $e$  are examined one by one. Those not dominated by  $S$  are inserted into the heap. The algorithm terminates when the heap is empty.

Figure 4 shows the heap contents for the example in Figure 2 while applying BBS. The second field in an element is a value of  $minDist$  where  $x+y$  is the distance function.

Action	Heap Contents	Skyline Points	Heap Size
initial state	$(e_0, 2)$	$\emptyset$	1
expand $e_0$	$(e_2, 4), (e_1, 7)$	$\emptyset$	2
expand $e_2$	$(e_5, 6), (e_7, 7), (e_1, 7), (e_6, 10)$	$\emptyset$	4
expand $e_5$	$(e_7, 7), (e_1, 7), (g, 8), (i, 8), (h, 10), (e_6, 10)$	$\emptyset$	6
expand $e_7$	$(e_1, 7), (g, 8), (i, 8), (l, 9), (h, 10), (e_6, 10), (n, 11), (m, 12)$	$\emptyset$	8
expand $e_1$	$(e_3, 7), (g, 8), (i, 8), (l, 9), (h, 10), (e_6, 10), (n, 11), (e_4, 12), (m, 12)$	$\emptyset$	9
expand $e_3$	$(g, 8), (i, 8), (l, 9), (c, 10), (h, 10), (e_6, 10), (a, 11), (b, 11), (n, 11), (e_4, 12), (m, 12)$	$\{g, i, a, n\}$	11

**Fig. 4.** BBS: Heap Contents

It has been proved in [12] that BBS is optimal in terms of I/O cost; that is, only “necessary” entries in the  $R$ -tree are visited and they are visited for only once.

To accelerate the dominance checking, the skyline points in the set  $T$  are indexed by a main-memory  $R$ -tree.

### 2.3 Divide-Conquer Skyline

The algorithm Divide-Conquer Skyline (DCSkyline) [9] applies a divide-conquer paradigm on the dataset.

Suppose that a rectangular region  $R$  contains some retrieved entries, in which an entry  $e_{min}$  has the minimal  $minDist$ . Divided from the lower-left corner of  $e_{min}$ ,  $R$  is partitioned into the 4 rectangular regions for further consideration:  $R_0$ ,  $R_1$ ,  $R_2$ , and  $R_3$ .  $R_3$  is the extent of the entry  $e_{min}$ .

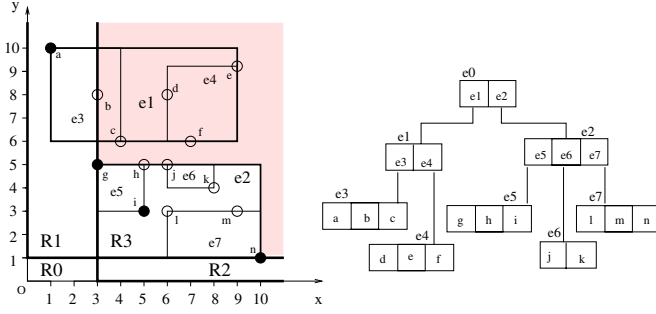


Fig. 5. Example of DCSkyline

Algorithm DCSkyline recursively applies a divide-conquer paradigm by the following ordering:

1. Find the skyline in  $R_1$  by calling DCSkyline, and return the lowest point  $p_1$ .
2. Find the skyline in  $R_2$  by calling DCSkyline, and return the leftmost point  $p_2$ .
3.  $R'_3 := R_3 - DR(p_1) - DR(p_2)$ , and find the skyline in  $R'_3$  by calling DCSkyline.

### 3 A New Skyline Query Algorithm

In this section, we propose a new algorithm Depth-First Skyline (DFS) for progressively processing skyline queries. We aim to speed up the computation time and reduce the memory space. This section firstly presents the motivation, followed by the description of the new algorithm DFS. Finally the analysis of DFS is presented.

#### 3.1 Motivation

The algorithm BBS is based on the best-first search technique for the nearest neighbour query. To avoid unnecessary I/O while retain the distance minimization, the  $R$ -tree entries on all possible paths are kept in the memory. An extreme example is illustrated in Figure 6, when all the points are having the same distance to the origin, the whole  $R$ -tree has to be read in before the first skyline point is retrieved. Thus, in this example, the space requirement is  $O(N)$ , where  $N$  is the scale of the dataset.

DCSkyline is efficient and scalable in terms of the memory usage, however it is not sensitive to the users' preference. Therefore, it is not a good progressive technique.

In this paper, we aim to develop a progressive skyline algorithm such that it is I/O optimal but with a much smaller memory space than BBS. Also a user's preference can be addressed.

#### 3.2 Depth-First Skyline

In this section, we present a new skyline algorithm Depth-First Skyline (DFS), which traverses the  $R$ -tree adopting a depth-first search paradigm.

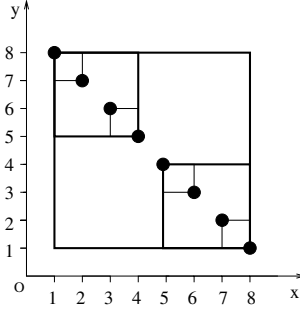


Fig. 6. Points with Same Distance

For example in Figure 2, consider the situation when the entry  $e_2$  is visited. Following the depth-first schema, it is unnecessary to expand  $e_1$  or any child of  $e_1$  before all skyline points in the entry  $e_2$  are retrieved. Subsequently, skyline points in  $e_5$  can be found without considering  $e_6$  or  $e_7$ . Thus only after visiting the path  $e_0$  to  $e_2$  to  $e_5$ , the algorithm DFS returns the first skyline points  $g$  and  $i$  to the user. Note that these two points are the “best” in terms of the distance function  $x + y$ .

In DFS, after an  $R$ -tree entry is visited, its child entries have the highest priority to be visited, unless this sequence increases the I/O cost. To bound the I/O cost, an entry  $e_i$  that is dominated by the lower-left corner of another entry  $e_j$  should be visited after  $e_j$ .

An overview of the algorithm is shown as follows.  $S$  is the skyline list.

---

#### Algorithm 1 DFS

---

**Input:**  $R$ -tree  $R$ .

**Output:** Skyline of the dataset.

**Description:**

```

1: Insert the root  $R$  into the entry list  $H$ ;  $S := \emptyset$ ;
2: while  $H \neq \emptyset$  do
3:   delete the first element  $e$  from  $H$ ;
4:   if  $e$  is not dominated by a point in  $S$  then
5:     if  $e$  is a data point then
6:        $S := S + \{e\}$ ; // output  $e$  as a skyline point
7:     else
8:        $expand(e)$ ; // see Entry Expanding
9:     end if
10:  end if
11:  filter  $S$ ; // when  $d = 2$ , see Skyline Points Stored
12: end while

```

---

**Entry Expanding.** In DFS, the  $R$ -tree entries to be visited are stored in an entry list. At each time, the first entry in the list is visited. Clearly for a depth-first search, the list should be sorted on decreasing order of the depth of the entries.

Heuristically, we expect that entries on lower levels are inserted to the list as forward as possible. Based on the observations in the previous parts, in DFS when expanding an entry  $e$ , the child entries should be checked individually. If a child entry is dominated by the lower-left corner of some other entries, it has to be inserted after all these entries. Otherwise, it should be inserted in front of the entry list, which is temporarily stored in the list  $C$  in Algorithm 2.

**Dominance Checking.** Since the dominance relationship between two intermediate entries (MBBs) specified in section 2.1, in the algorithm DFS, entries in the entry list can be filtered out by both the already-found skyline points and another intermediate entry.

---

**Algorithm 2 Expand( $e$ )**


---

**Input:** An intermediate entry  $e$  of  $R$ -tree, skyline point set  $S$ , entry list  $H$ .

**Output:** Changed entry list  $H$ .

**Description:**

```

1: retrieve child entries from  $e$ ;  $C := \emptyset$ ;
2: pick child entries  $e_{1..k}$  not dominating each other; //see Dominance Checking
3: for  $i \in \{1..k\}$  do
4:   if  $e_i$  is dominated by a point in  $S$  or an entry in  $H$  //see Dominance Checking
     then
5:     discard  $e_i$ ;
6:   else if  $e_i$  is dominated by the lower-left corner of an entry in  $H$  then
7:     insert  $e_i$  after the last such entry;
8:   else
9:      $C := C \cup \{e_i\}$ ;
10:  end if
11: end for
12: sort  $C$  on increasing order of  $\min Dist$ ;
13:  $H := C \cup H$ ;

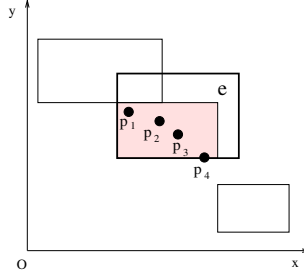
```

---

When to eliminate the dominated entries is also important for minimising the storage size. In the algorithm DFS, the “forward checking” technique is applied, which tries to prevent dominated entries from being stored in the entry list. Firstly, when expanding an entry, the child entries are compared so that they are not dominating each other. In the example in Figure 2, after expanding  $e_2$ , the three child points  $e_5$ ,  $e_6$  and  $e_7$  are compared, and  $e_6$  is pruned immediately. Secondly, a newly inserted entry is always checked so that it is not dominated by any other entry in the list at that time.

**Skyline Points Stored.** In the 2-dimensional space, the memory usage can be further reduced, observing that it is not necessary to maintain all the skyline points found already in memory for dominance checking.

See Figure 7 for illustration.  $e[x^l, x^h] \times [y^l, y^h]$  is the first entry in the entry list  $H$ . The shaded region  $SR[x^l, x] \times [y^l, y]$  is the largest rectangle, such that region  $[0, x] \times [0, y]$  only intersects with entry  $e$ . Based on the depth-first schema, skyline points in



**Fig. 7.** Stored Skyline Points in 2D Space

$SR$  are found without considering other entries. After finding the skyline points in  $SR$ , only  $p_1$  and  $p_4$ , the highest and the rightmost points, should be stored in memory for dominance checking. This is because that the dominance region of  $p_1$  and  $p_4$  contains the dominance region of all the other skyline points between them. Formally,  $(DR(p_1) \cup DR(p_4)) \supseteq DR(p_i), i = 1..4$ .

In a higher dimensional space, since the dominance region of the skyline points may not be contained by each other, they should be maintained in memory for further dominance checking.

**An Example.** Figure 8 shows the entry list contents for the example in Figure 2 while applying DFS. The distance function is still  $x + y$  for computing  $minDist$ .

Action	List Contents	Skyline Points	List Size
initial state	$(e_0, 2)$	$\emptyset$	1
expand $e_0$	$(e_2, 4), (e_1, 7)$	$\emptyset$	2
expand $e_2$	$(e_5, 6), (e_7, 7), (e_1, 7)$	$\emptyset$	3
expand $e_5$	$(g, 8), (i, 8), (e_7, 7), (e_1, 7)$	$\{g, i\}$	4
expand $e_7$	$(n, 11), (e_1, 7)$	$\{g, n\}$	2
expand $e_1$	$(e_3, 7)$	$\{g, n\}$	1
expand $e_3$	$(a, 11)$	$\{n, a\}$	1

**Fig. 8.** DFS: List Contents

### 3.3 Analysis

It can be easily proved that the algorithm DFS is correct, and is optimal in terms of I/O cost. Due to the space limit, we omit the proofs of the following theorems[10].

**Theorem 1.** *Points progressively produced by DFS form the skyline of the given dataset.*

**Theorem 2.** *Suppose that  $e$  is an entry accessed by DFS. Then  $e$  intersects  $SSR$ .*

Since it is complicated to theoretically analysis the storage requirement of DFS on a high dimensional space or when there exists much overlapping between  $R$ -tree entries,

we will show the performance in the experiments in the next section. Here we only prove that the storage is logarithmic in a specific situation.

**Lemma 1.** *In a 2D space, the maximal memory requirement of the algorithm DFS is  $O(\log N)$  in the worst case, where  $N$  is the size of the dataset, if there is no intersection in the index  $R$ -tree.*

**Proof:** First consider the length of the entry list. As described in section 3.2 when expanding an entry, a child entry  $e_i$  is either attached in front of the entry list following a depth-first search schema, or inserted after another entry in the list  $e_j$  whose lower-left corner dominating  $e_i$ . Since there is no intersection in the index  $R$ -tree, the lower-left corner of  $e_j$  dominating  $e_i$  follows that  $e_j$  dominating  $e_i$ , and  $e_i$  is eliminated immediately. Thus the searching of the  $R$ -tree is a depth-first search, which needs a logarithmic space.

It is clear that the memory space for the stored skyline points is also  $O(\log N)$ . Thus the total memory space used in DFS is  $O(\log N)$ .  $\square$

## 4 Performance Study

In the experiments, “independent” and “anti-correlated” datasets are used. Points in the independent datasets are uniformly distributed, while points in anti-correlated datasets that are good at one dimension are tend to be bad in at least one other dimension [7]. The index tree is  $R^*$ -tree with 200 entries per page. All experiments are run on a 1.8G CPU with 512M Ram.

### 4.1 Two Dimensional Space

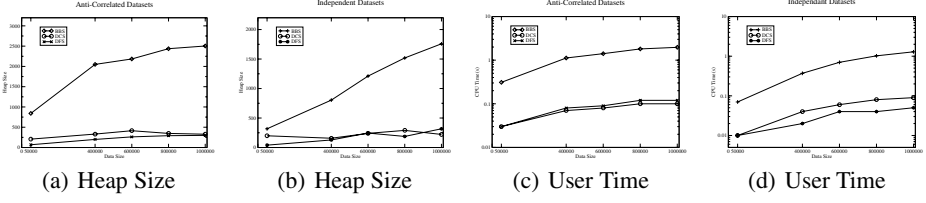
Figure 9 shows the maximal storage and the user time of the three algorithms on the 2-dimensional space. *DFS* performs the best.

### 4.2 Multi-dimensional Space

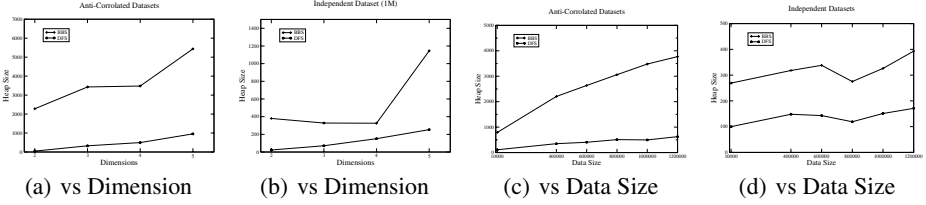
Datasets on the space from 2 to 5 dimensions are used. Two algorithms are compared in the experiments. Note that here in *DFS*, all the skyline points are stored in memory when  $d = 2$  for comparison.

Figure 10 (a) and (b) shows the maximal storage required by these algorithms on different dimensional spaces with 1M data points. While in Figure 10 (c) and (d), the performance on datasets with different number of points are compared when all datasets are on a 4 dimensional space. With different scale and different dimensionality, *DFS* needs much less storage than *BBS*.

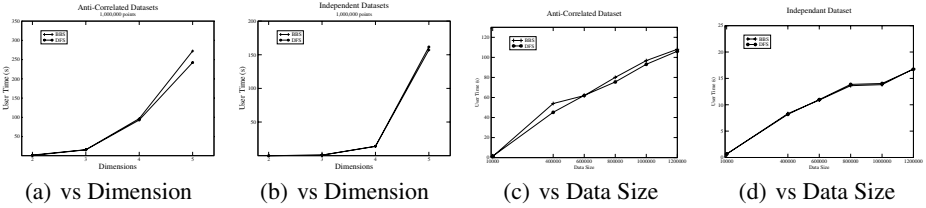
Figure 11 compares the user time on different conditions, which shows that the time complexity are similar. The dataset used is the same as in the Figure 10. The reason is that *DFS* has a relatively high CPU cost overhead when storing a new entry into the list, since the whole list should be read; on the other hand, *BBS* needs to compare much more entries with the already-found skyline.



**Fig. 9.** 2-Dimensional Space



**Fig. 10.** Maximal Storage on High Dimensional Space



**Fig. 11.** User Time on High Dimensional Space

### 4.3 Progressive Performance

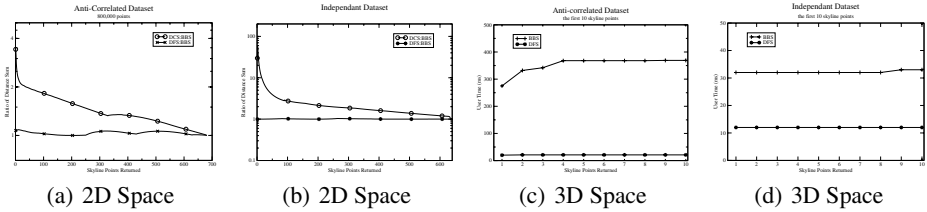
To study the progressive performance on the 2-dimensional space, we summarise the  $k$  distances from the origin to the first  $k$  skyline points retrieved. The ratio of the sums between different algorithms are shown in Figure 12 (a) and (b). Note that in algorithm *BBS*, the sum is minimised. The performance of *DFS* is much better than *DCS* (DCS), as a result of the different retrieving sequence.

Figure 12 (c) and (d) shows the time to retrieve the first 10 skyline points on the 4-dimensional space using the same dataset as in Figure 11 (c) and (d). *DFS* still works much better, since *DFS* works like a depth-first search algorithm.

## 5 Conclusion

In this paper, we develop the algorithm *DFS* that are I/O cost optimal for the skyline queries. The experiments showed that the techniques used have a great impact on improving the performance of skyline query algorithms. The new algorithm works on high dimensional space, and is sensitive to the user's preference.





**Fig. 12.** Progressive Performance

In the future, skyline queries based on other kind of index trees will be studied since  $R$ -tree is not efficient on a higher dimensional space when  $d > 5$ .

## References

1. S. Borzsonyi, D. Kossmann and K. Stocker. "The Skyline Operator", *International Conference on Data Engineering ICDE*, 2001.
2. N. Beckmann, H. Kriegel, R. Schneider, B. Seeger. "The R\*-tree: An Efficient and Robust Access Method for Points and Rectangles", *SIGMOD*, 1990.
3. H. Ferhatosmanoglu, I. Stanoi, D. Agrawal, A. Abbadi. "Constrained Nearest Neighbor Queries", *SSTD*, 2001.
4. V. Gaede and O. Gunther. "Multidimensional Access Methods", *Computing Surveys*, 30(2): 170-231, 1998.
5. G. Hjaltason, H. Samet. "Distance Browsing in Spatial Databases", *ACM TODS*, 24(2):265-318, 1999.
6. H.T. Kung, F. Luccio and F.P. Preparata. "On finding the maximum of a set of vectors", *Journal of the ACM*, 22(4):469-476, 1975.
7. D. Kossmann, F. Ramsak and S. Rost. "Shooting Stars in the Sky: An Online Algorithm for Skyline Queries", *VLDB'02*, 2002.
8. Marios Hadjieleftheriou. "Spatial Index Library", <http://www.cs.ucr.edu/~marioh/spatialindex/index.html>, 2002.
9. H. X. Lu, Y. Luo and X. Lin. "An Optimal Divide-Conquer Algorithm for 2D Skyline", *ADBIS*, 2003.
10. Y. Luo, H. X. Lu and X. Lin. "A Scalable and I/O Optimal Skyline Processing Algorithm (full paper)", 2004.
11. F.P. Preparata and M.I. Shamos. "Computational Geometry: An Introduction", *Springer-Verlag, New York, Berlin, etc.*, 1985.
12. D. Papadias, Y. Tao, G. Fu, B. Seeger. "An Optimal Progressive Algorithm for Skyline Queries", *SIGMOD*, 2003.
13. N. Roussopoulos, S. Kelly, F. Vincent. "Nearest Neighbor Queries", *SIGMOD*, 1995.
14. R. Steuer "Multiple Criteria Optimization", *Wiley New York*, 1986.
15. K.L. Tan, P.K. Eng and B.C. Ooi. "Efficient Progressive Skyline Computation", *VLDB*, 2001.

# Linearization Approach for Efficient KNN Search of High-Dimensional Data

Zaher Al Aghbari<sup>1</sup> and Akifumi Makinouchi<sup>2</sup>

<sup>1</sup> Computer Science Department, University of Sharjah  
P.O.Box 27272, Sharjah, UAE  
zaher@sharjah.ac.ae

<sup>2</sup> Graduate School of Information Science and EE, Kyushu University  
Higashi-ku, hakozaiki 6-10-1, Fukuoka, Japan  
akifumi@is.kyushu-u.ac.jp

**Abstract.** K-Nearest Neighbors (KNN) search in high-dimensional feature spaces is an important paradigm for a variety of applications. Despite the continuous efforts in the past years, algorithms of data partitioning methods (DPMs), such as R\*-tree, SR-tree and self-organizing maps (SOM), to find the exact KNN answer set at high dimensions are outperformed by a linear scan method. In this paper, we present a “plug&search” method to greatly speed up the exact KNN search of existing DPMs. The idea is to linearize the data partitions produced by a DPM, rather than the points themselves, based on the distances between the partitions and a selected reference point, thus resulting in a one-dimensional array-index, that is simple, compact and yet fast index structure. Our KNN search algorithm sets conditions to skip irrelevant partitions and early stop the search as soon as the exact KNN answer points are retrieved.

## 1 Introduction

An important paradigm in recent database applications (i.e. multimedia databases [13], time series databases [11], data mining [6], etc.) is the content based retrieval of similar objects, where an object is represented by a point in a  $d$ -dimensional feature space. The problem of finding the exact, or approximate, KNN points in a database to a given query point has been addressed by a large number of works. These related works can be classified into four approaches:

**Space/Data Partitioning Approaches** like grid-file [24] and K-D-tree [2] partition the data space into disjoint regions through cuts on some dimensions regardless of data clusters. On the other hand, data partitioning methods (DPMs) like R\*-tree [1], SS-tree [18], SR-tree [10] and X-tree [3] divide the data space according to the distribution of points in these indexing trees.

**Dimensionality Reduction Approaches** such as [9][7][5] apply “dimensionality reduction” techniques on the data and then insert the data into the indexing trees, which are known to perform well on low-dimensional data.

**Scanning-Based Approaches** such as the VA-file [17] and LPC-file [4] divide the data space into  $2^b$  rectangular cells, where  $b$  denotes a user specified number of bits. Each cell is allocated a bit-string of length  $b$  that approximates the data points that fall into a cell. The KNN search starts by scanning the entire file of approximations and filtering out the irrelevant points based on their approximations

**Linearization Approaches**, such as space-filling curve methods (Z-order curve [15] or Hilbert curve [8]) map  $d$ -dimensional points into a one-dimensional space (curve), i.e.  $\mathcal{R}^d \rightarrow \mathcal{R}^1$ . As a result, one can issue a range query along the curve to find nearest neighbors.

In this paper, we propose a new “plug&search” KNN search method, called *array-index*, that speeds up the KNN search and preserves the properties of the underlying DPM. The proposed method reads the data partitions produced by a high-dimensional DPM and then it computes the *representative vector*  $\mathbf{m}_i$  of each data partition. A reference point  $\mathbf{R}$  is selected and the distance  $D(\mathbf{m}_i, \mathbf{R})$  between each representative vector  $\mathbf{m}_i$  and  $\mathbf{R}$  is computed. Using the computed distances, we map the data partitions into a 1-dimensional distance space, called *array-index*, in which the mapped data partitions are sorted based on their similarity to  $\mathbf{R}$ . Using a small sorted array structure allows us to find the most similar data partition, called a *winner partition* (WP), to a given  $\mathbf{q}$  using the fast binary search algorithm whose complexity is  $O(\log_{N_c})$ . Our KNN search algorithm, starts by considering the points in the WP for possible KNN answers. Then, the search proceeds along the *array-index* by checking the points in the partitions to the left and right of WP. During traversal of the *array-index*, the algorithm prunes irrelevant partitions and it stops if checking new data partitions do not result in any new KNN answer points on both sides. This method retrieves *exact* KNN answer points to a given a query point.

The rest of this paper is organized as follows: Section 2 discusses how to linearize the data partitions and build the *array-index*. Section 3 presents the KNN search algorithm. The experiments are discussed in Section 4. Finally, we conclude the paper in Section 5.

## 2 Linearizing the Data Partitions

To improve the KNN search performance of the DPMs on real data (highly skewed and correlated), we propose a plug&search method, called *array-index*. The *array-index* utilizes the partitions (leaf nodes or clusters) of any of the known DPMs and linearizes these partitions by ordering them based on their distances from a selected reference point, thus resulting in a 1-dimensional *array-index*. Hence, similar partitions are placed next to each other; enabling us to tailor a fast algorithm to find the KNN answers points. That is, given  $\mathbf{q}$ , the algorithm only traverses a small range (to the left and right of  $\mathbf{q}$ ) that bounds the  $K$  answer set.

## Linearization

Linearization is achieved by selecting a reference point  $\mathbf{R}$  and then ordering all partitions according to their distances from the selected  $\mathbf{R}$ . We defined three types of reference points:

- (1) *center*  $\mathbf{R}$ , which is the average of all partitions' representative vectors,
- (2) *edge*  $\mathbf{R}$ , which is the farthest partition from *center*  $\mathbf{R}$ , and
- (3) *random*  $\mathbf{R}$ , which is a randomly selected partition.

After selecting  $\mathbf{R}$ , we compute the Euclidean distance  $D(\mathbf{m}_i, \mathbf{R})$  between the representative vector  $\mathbf{m}_i$  of each partition and  $\mathbf{R}$ . Thus, we map the partitions into a one-dimensional *array-index* space in which similar partitions are placed close to each other. Partitions that are equi-distant from  $\mathbf{R}$ , which may *not* be similar, are placed on the same position in the 1-dimensional space. However, these equi-distant partitions are pruned during the KNN search.

The computed distances  $D(\mathbf{m}_i, \mathbf{R})$  are inserted into the *array-index*. Along with each distance, a pointer to the *partition-point-list* of the corresponding partition is set. The *partition-point-list* of partition  $C_i$  is a list of the points that are associated with  $C_i$  and these points are sorted based on their Euclidean distances from the representative vector  $\mathbf{m}_i$  of  $C_i$ . Hence, the *array-index* contains a list of partitions sorted based on their distances from  $\mathbf{R}$  and each partition contains a pointer to a list of points sorted based on their distances from  $\mathbf{m}_i$ . Also, for every  $C_i$ , we compute its radius  $r_i$  that reflect the size of  $C_i$ . The constructed *array-index* is very small in size since it contains a number of cells equal to the number of generated partitions and every cell contains three fields: distance, pointer and radius.

## Search Conditions

Here we discuss the pruning and ending conditions that are used to control the traversal of the *array-index* and speed up the KNN search. Let  $P$  be a set of points in  $C_i$ , then radius  $r_i$  is the largest distance between the representative vector  $\mathbf{m}_i$  of  $C_i$  and any point. That is,

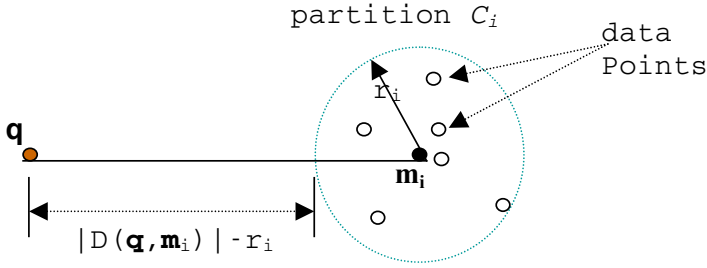
$$r_i = \max \{ D(\mathbf{m}_i, \mathbf{p}) \mid \mathbf{p} \in P \} \quad (1)$$

If the set of radii of all partitions is  $S$ , then the maximum radius  $r_{max}$  is:

$$r_{max} = \max \{ r_i \mid r_i \in S \} \quad (2)$$

## Partition Pruning

Employing a pruning strategy is necessary due to the fact that mapping partitions based on their Euclidean distances from  $\mathbf{R}$  places equi-distant partitions, which may not be similar, onto the same position in the 1-dimensional distance space. Let  $D_{knn}^c$  be the current largest distance in the *KNN\_list*, where the *KNN\_list* keeps a sorted list



**Fig. 1.** Radius of a partition.

of the  $K$  most similar points to  $\mathbf{q}$ . Hence, partition  $C_i$  is pruned if the minimum distance between  $\mathbf{q}$  and  $C_i$  is greater than  $D_{knn}^c$  (see Figure 1).

**Condition 1:** During the KNN search, partition  $C_i$  is pruned if the following condition holds true:

$$|D(\mathbf{q}, \mathbf{m}_i)| - r_i > D_{knn}^c \quad (3)$$

### Partition Ending

Since the *array-index* stores the partitions sorted in an ascending order based on their distances from  $\mathbf{R}$ , it is possible to traverse only a small range (left and right of  $\mathbf{q}$ ) that bounds the answer set by utilizing an ending condition. To guarantee that no qualifying points are missed, the search should not end if there is a partition overlapping  $D_{knn}^c$  because there might be a point, in the overlapped portion of the partition, whose distance from  $\mathbf{q}$  is smaller than  $D_{knn}^c$ . To check whether there exists a partition overlapping with  $D_{knn}^c$ , we need to check all partitions whose  $\mathbf{m}_j$  are in the vicinity of  $D_{knn}^c$  up to  $r_{max}$  (that is  $D_{knn}^c + r_{max}$ ) because  $r_{max}$  specifies the position of the farthest  $\mathbf{m}_j$  of  $C_j$  that might be overlapping with. Formally, we state the partition ending condition of the KNN search as follows:

**Condition 2:** The KNN search ends when the following condition holds true in both the left and right directions:

$$D_{knn}^c + r_{max} < |D(\mathbf{R}, \mathbf{m}_i)| - |D(\mathbf{R}, \mathbf{q})| \quad (4)$$

### Point Pruning

The *array-index* method sorts the points associated with a partition in an ascending order such that the point with the smallest distance from  $\mathbf{m}_i$  is closest to  $\mathbf{m}_i$  and the point with the largest distance from  $\mathbf{m}_i$  is farthest from  $\mathbf{m}_i$ . That is the points are mapped into a one-dimensional distance space. When sorting the points associated with a partition, equi-distant points fall into the same position in the one-dimensional

distance space. As partition  $C_i$  is visited, if  $\mathbf{q}$  is inside  $C_i$ , then all the points within  $C_i$  are checked. On the other hand, if  $\mathbf{q}$  is outside  $C_i$ , the search within  $C_i$  starts by computing  $|D(\mathbf{q}, \mathbf{p})|$  of the farthest point  $\mathbf{p}$  from  $\mathbf{m}_i$  (closest point  $\mathbf{p}$  to  $\mathbf{q}$ ) and then inserts  $\mathbf{p}$  into  $KNN\_list$  if  $|D(\mathbf{q}, \mathbf{p})| < D_{knn}^c$ . Then, the search proceeds to the next point and so on. Equi-distant points whose distance  $|D(\mathbf{q}, \mathbf{p})| \geq D_{knn}^c$  are pruned. Formally,

**Condition 3:** During the KNN search in partition  $C_i$ , a point  $\mathbf{p}$  is pruned if the following condition holds true:

$$|D(\mathbf{q}, \mathbf{p})| \geq D_{knn}^c \quad (5)$$

### Point Ending

Since the points are sorted in  $C_i$  based on their distances from  $\mathbf{m}_i$ , it is possible to stop checking the points if further checking would not result in any new KNN answer points. Formally,

**Condition 4:** The KNN search in partition  $C_i$  ends if the following condition holds true:

$$|D(\mathbf{q}, \mathbf{m}_i)| - |D(\mathbf{m}_i, \mathbf{p})| \geq D_{knn}^c \quad (6)$$

## 3 KNN Search Algorithm

Here, we present a *KNN\_Search* algorithm for the *array-index* to find the exact KNN points to a given query  $\mathbf{q}$ . Let's first explain some of the routines and notations in the *KNN\_Search* algorithm. The *KNN\_list* keeps the distances of, and pointers to, the  $K$  most similar points to  $\mathbf{q}$ . The *update\_KNN\_list( $C_i$ )* routine, which is shown below, reads  $C_i$  from disk (line 1) and processes the sorted points in  $C_i$  starting from the point that is farthest from  $\mathbf{m}_i$ . The algorithm (line 3) checks whether the point ending condition (Condition 4) holds true, if it does, the point traversal is ended. Then, in line 4, it checks if the point pruning condition (Condition 3) holds true, that is if the current point is an equi-distant point, if it does, the current point is skipped. If both Conditions 3 and 4 do not hold true, then  $\mathbf{p}_i$  is inserted into the *KNN\_list* based on its distance  $D(\mathbf{q}, \mathbf{p}_i)$ .

**Algorithm:** *Update\_KNN\_list( $C_i$ )*

1. read( $C_i$ )
2. For  $\mathbf{p}_i$ , where  $i = 1, 2, \dots$  number of points in  $C_i$ .
3. If (Condition 4) then break and go to step 6
4. Else If (Condition 3) then skip  $\mathbf{p}_i$  and go to step 2
5. Else insert  $\mathbf{p}_i$  into *KNN\_list*
6. return

Lptr and Rptr are pointers used to traverse the *array-index* toward the left and the right of the winner partition WP, respectively. The first essential step in the *KNN\_Search* algorithm (shown below) is to find the WP, which is a partition whose  $\mathbf{m}_i$  is the most similar to  $\mathbf{q}$ , then the *KNN\_list* is updated with the points associated with the WP (line 1). Line 2 initializes Lptr and Rptr to equal the WP pointer. Line 3 defines Boolean variables that indicate the flow of the KNN search. The search for KNN points is contained in the loop between lines 4-23. Lines 5-8 traverse the *array-index* towards the left of WP if the left traversal has not been ended. The distance between  $\mathbf{q}$  and the  $\mathbf{m}_{Lptr}$  of the next unvisited partition to the left of WP is computed (line 6). Then, in line 7, the algorithm checks if the partition ending condition (Condition 2) for the left traversal holds true, if it does, the left traversal is ended. If the left traversal is not ended, the currently visited partition is checked against the partition pruning condition (Condition 1), if it holds true, the currently visited partition is pruned and the algorithm advances to the next unvisited partition in left direction.

Similarly, lines 9-12 traverse the *array-index* towards the right. Lines 14-15 check if both traversal directions (left and right) have been ended. If so, the search ends and then the *KNN\_list* is returned. In lines 16-19, if the left traversal and right traversal have not been ended, the *KNN\_list* is updated with one of the two currently visited partition (the one with the smaller distance from  $\mathbf{q}$ ), then the traversal in the opposite direction is blocked temporarily, so that the next iteration of the algorithm fetches the next partition in the direction of the *closer* partition (the one with the smaller distance). Lines 20-21 handle the case when the left traversal is ended and the search is advancing only in the right direction, thus, the *KNN\_list* is updated with the next qualifying partition on the right direction. Similarly, in lines 22-23, the search only advances in the left direction. In line 24, the *KNN\_list* is returned with the exact KNN answer points.

**Algorithm:** *KNN\_Search*

**Input:** query  $\mathbf{q}$  and the *array-index*

**Output:** *KNN\_list*

1.  $WNptr = find\_WN(\mathbf{q})$ ; and  $update\_KNN\_list(WNptr)$
2.  $Lptr = Rptr = WNptr$
3.  $KnnFound = FALSE$ ;  $TraverseLeft = TraverseRight = TRUE$
4. While (NOT *KnnFound*)
5.   If (left traversal is not ended AND *TraverseLeft*)
6.     compute  $|D_L(\mathbf{q}, \mathbf{m}_{Lptr})|$
7.     If (Condition 2) then end left traversal; go to step 9
8.     If (Condition 1) then prune  $C_{Lptr}$ ; go to step 5
9.   If (right traversal is not ended AND *TraverseRight*)
10.    compute  $|D_R(\mathbf{q}, \mathbf{m}_{Rptr})|$
11.    If (Condition 2) then end right traversal; go to step 13
12.    If (Condition 1) then prune  $C_{Rptr}$ ; go to step 9
13.   *TraverseLeft = TraverseRight = TRUE*
14.   If (left and right traversal are ended) then

```

15.   KnnFound = TRUE
16.   Else If (left and right traversal are not ended) then
17.     If ( $D_L < D_R$ ) then
18.       update_KNN_list( $C_{Lptr}$ ); TraverseRight = FALSE
19.     Else update_KNN_list( $C_{Rptr}$ ); TraverseLeft = FALSE
20.   Else If (only left traversal is ended) then
21.     update_KNN_list( $C_{Rptr}$ )
22.   Else If (only right traversal is ended) then
23.     update_KNN_list( $C_{Lptr}$ )
24.   return KNN_list

```

## 4 Experiments

We use a Haar wavelet transform to compute the color vectors of images. The color space used in this paper is the YIQ, where Y carries luminance information and I and Q carry color information and some luminance information. To demonstrate the efficiency of our KNN search algorithm, we generated 3 databases with different dimensionalities: 3, 12, and 48 dimensions by applying a 7-level, 6-level, and 5-level decompositions respectively, of the Haar wavelet transform on the image data. To show the applicability of the *array-index* to different DPMs, we partitioned each database by two different DPMs, *Tree-based partitioning* and *Nontree-based clustering*. Namely, we used the SR-tree and SOM. For details on SR-tree and SOM methods, please refer to [10] and [16].

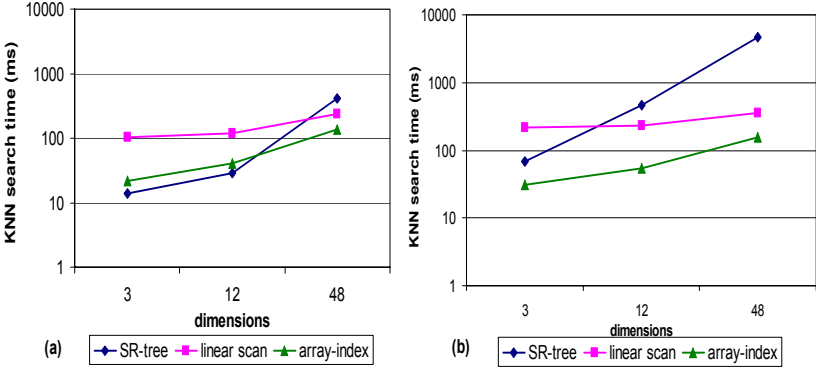
To investigate the KNN search efficiency of the *array-index*, we compare the KNN search time of the *array-index* with that of the underlying DPMs. That is, we compare: (1) SR-tree based *array-index* versus SR-tree, and (2) SOM based *array-index* versus SOM.

Figures 2 and 3 show the results of comparing KNN search times of the SR-tree based *array-index* with the SR-tree and SOM based *array-index* with the SOM, respectively, at different dimensions (3, 12 and 48) and different values of  $K$  (1 and 100). In these experiments, we used a database of 40000 images and the *edge R* to build both the SR-tree based *array-index* and SOM based *array-index*. Also, we set the SOM map size to 200x200. The results are the average of 10 randomly selected queries. We added the KNN search times of a linear scan method into these figures for referential comparisons. Notice that the scale of the y-axis of Figures 2 and 3 are logarithmic.

Figure 2.a shows that at low dimensions ( $d=3$ ) and small values of  $K$  ( $K=1$ ) the SR-tree outperforms both the linear scan and the SR-tree based *array-index*. That is expected since the indexing trees are known to perform well at low dimensions. However, as the value of  $K$  increases ( $K \geq 1$ ), even at low dimensions ( $d=3$ ), SR-tree needs to access more partitions, leading to an increase in KNN search time, hence the performance of the SR-tree degraded as compared to that of the *array-index*, although it remained better than that of the linear scan. At higher dimensions ( $\geq 12$ ) and large

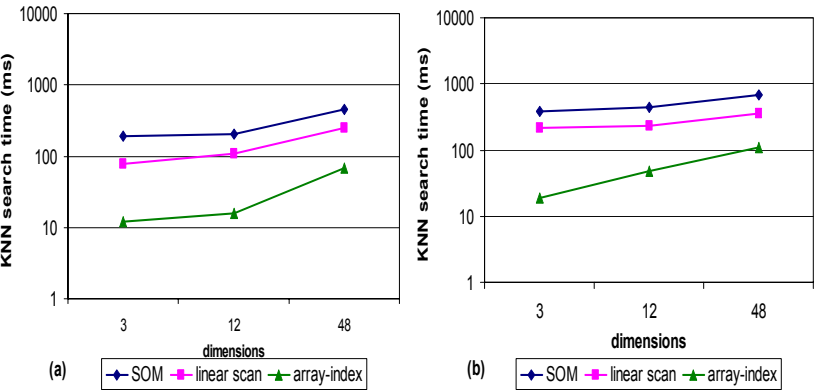


values of  $K$  ( $>1$ ), the *array-index* shows much faster KNN search time than both the linear scan and SR-tree.



**Fig. 2.** Comparing SR-tree with SR-tree based *array-index*: average KNN time versus number of dimensions (a)  $K = 1$  and (b)  $K = 100$ .

From Figure 3 we notice that the SOM based *array-index* performed much better than both the linear scan and SOM at all dimensions and all values of  $K$ . On the other hand, at all dimensions and all values of  $K$ , the SOM is slower than the linear scan by a factor of about 1.5. That is reasonable because the KNN search algorithm of SOM (explained above) requires to, first, check all clusters to determine the winner cluster, and then it visits all the clusters to check the points associated with the visited clusters. We conclude that plugging the *array-index* into a DPM greatly speeds up the KNN search time.



**Fig. 3.** Comparing SOM with SOM based *array-index*: average KNN time versus number of dimensions (a)  $K = 1$  and (b)  $K = 100$ .

## 5 Conclusion

We presented an efficient plug&search method, called *array-index*, to speed up the *exact* KNN search of a wide range of existing DPMs and at the same time preserves their properties. The *array-index* method linearizes the data partitions of the underlying DPM instead of the data points themselves; hence the *array-index* is a compact indexing structure. Although, in this paper, we used an image database as an example of multi-dimensional data, the *array-index* method can be adapted to any multi-dimensional data.

The compactness of the *array-index*, the ordering of partitions in the *array-index*, and the ordering of objects inside the partitions are the main reasons behind the fast KNN search time as compared to other known methods. That is, the compactness of the *array-index* enables us to put the whole index structure in main memory leading to an elimination of disk operations, the ordering of partitions reduces the number of disk access to retrieve the data pages, and the ordering of points inside the partitions reduces the number of distance calculations of the points in the retrieved data pages.

## References

1. N.Beckmann, H.Kriegel, R.Schneider, B.Seeger. *R\*-tree: An Efficient and Robust Access Method for Points and Rectangles*. ACM SIGMOD, pp. 322-331, May 1990.
2. J.L.Bentley *Multidimensional Binary Search Trees in Database Applications*. IEEE Trans. on Software Engineering, SE-5(4):333-340, July 1979.
3. S.Berchtold, D.Keim, H.P.Kriegel. *The X-tree: An Index Structure for High-Dimensional Data*. VLDB 1996.
4. G-H.Cha, X.Zhu, D.Petkovic, C-W.Chung. *An Efficient Indexing Method for Nearest Neighbor Searches in High-Dimensional Image Databases*. IEEE Trans. on Multimedia, Vol. 4, No. 1, March 2002.
5. K.Chakrabarti, S.Mehrotra. *Local Dimensionality Reduction: A New Approach to Indexing High Dimensional Space*. 26th VLDB, Egypt, 2000.
6. C.Faloutsos and K.Lin. *A Fast Algorithm for Indexing, Data-Mining and Visualization of Traditional and Multimedia Datasets*. ACM SIGMOD, May 1995.
7. C.Faloutsos, M.Ranganathan, Yannis Manolopoulos. *Fast Subsequence Matching in Time-Series Databases*. ACM SIGMOD, 1994.
8. C.Faloutsos, S.Roseman. *Fractals for Secondaru Key Retrieval*. 8th ACM Symposium on Principles of Database Systems (PODS), March 1989.
9. M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Perkovic, D. Steele, P. Yanker. *Query by Image and Video Content: The QBIC System*. IEEE, Sept. 1995.
10. N.Katayama, S.Satoh. *The SR-tree: An Index Structure for High-Dimensional Nearest Neighbor Queries*. ACM SIGMOD, May 1997.
11. E.Keogh, K.Chakrabarti, S.Mehrotra, M.Pazzani. *Locally Adaptive Dimensionality Reduction for Indexing Large Time Series Databases*. ACM SIGMOD, 2001.
12. J.M.Kleinberg. *Two Algorithms for Nearest Neighbor Search in High Dimensions*. 29th ACM Symposium on Theory of Computing, 1997.

13. P. P.Korn, N.Sidiropoulos, C.Faloutsos, E.Siegel, Z.Protopapas. *Fast and Effective Retrieval of Medical Tumor Shapes*. IEEE Trans. on Knowledge and Data Engineering, Vol.10, No.6. Nov/Dec 1998.
14. J.Nievergelt, H.Hinterberger, K.Sevcik. *The gridfile: An Adaptable Symmetric Multikey File Structure*. ACM Trans. on Database Systems, 9(1):38-71, 1984.
15. J.Orenstein *Spatial Query Processing in an Object-Oriented Database System*. Proc. ACM SIGMOD, May 1986.
16. A.Rauber, J.Paralic, E.Pampalk. *Empirical Evaluation of Clustering Algorithms*. Journal of Information and Organizational Sciences, Vol. 24, No. 2, pp. 195-209, 2000.
17. R.Weber, H-J.Schek, S.Blott. *A Quantitative Analysis and Performance Study for Similarity-Search Methods in High-Dimensional Spaces*. Proc. of the 24th VLDB, USA, 1998.
18. D.A.White, R.Jain. *Similarity Indexing with the SS-tree*. Proc. of Int'l Conference on Data Engineering (ICDE), 1996.

# SQL/MDR: Query Language for Consistent Access of Metadata Registries Based on SQL3

Dongwon Jeong<sup>1</sup>, Young-Gab Kim<sup>2</sup>, and Doo-Kwon Baik<sup>2</sup>

<sup>1</sup> Research Institute of Information and Communication Technology, Korea University,  
1, 5-ka, Anam-dong, Sungbuk-gu, Seoul, 136-701, Korea  
withimp@korea.ac.kr

<sup>2</sup> Dept. of Computer Science & Engineering, Korea University,  
1, 5-ka, Anam-dong, Sungbuk-gu, Seoul, 136-701, Korea  
{ygkim, baik}@software.korea.ac.kr

**Abstract.** This paper proposes a query language to consistently access metadata registries. In current, many metadata registries have been built in various fields. Because there is no access method to access the metadata registries in a standard manner, many management systems for them have been developed in various and different access methods. In this paper, we propose a metadata registry query language that allows us to access consistently all metadata registries in a standardized manner. The query language, SQL/MDR is an extension of SQL3, which is familiar to existing database managers. Consequently, the query language reduces the development cost of a metadata registry management system. And it also enables all metadata registries to be accessed in a consistent manner.

## 1 Introduction

ISO/IEC JTC 1 developed ISO/IEC 11179 to enhance interoperability of databases. The main concept of this standard is the data element, which is a set of attributes such as identification, representation, and allowable value of data, and is a minimal unit to specify data. A metadata registry is the set of data elements and is one of the key elements of ISO/IEC 11179: [1], [15]

Because of its advantages such interoperability and dynamic metadata management, until now, many metadata registries in various fields have been built to efficiently manage data. The representative examples of the metadata registries are as follows:

- KISTI (Bibliographic metadata registry): [2]
- EPA (Environmental Data Registry): [4], [10]
- NHIK (Australian National Health Information Knowledgebase): [5], [11]
- U.S. ITS (U.S. Intelligent Transportation System): [6], [12]

The built metadata registries are based on the standard, ISO/IEC 11179. Thus, there are similar access patterns to handle the metadata registries. ISO/IEC 11179 provides no standard metadata registry access method. Until now, no research on the

standard access method has been made. It causes several problems such as duplicate developments of the access method, metadata registry access in different manners, etc. Most of all, even though metadata registries are created according to the standard, there is no standard access method to consistently access them.

This paper proposes a metadata registry query languages as the standard access method to solve the issues aforementioned. The metadata query language is referred as SQL/MDR and is an extension of SQL3, which is the international standard query language.

To achieve this goal, we first analyzed and defined query patterns. SQL/MDR is designed based on the query patterns and extended from SQL3: [13], [14]. SQL/MDR is the metadata registry query language to access the metadata registries in a standardized manner. Therefore, it reduces time and effort for developing systems. It enables all metadata registries to be accessed consistently, thus we can increase interoperability between metadata registry systems.

## 2 Operation Analysis

### 2.1 Metadata Registry Operations

The basic operations for managing metadata registry are classified into data element search, grouping element search, data element registration, registration status management, version management of data element, and so on. Metadata registries are managed and extended by these operations.

The data element search is a function to retrieve a proper data element. A data element has many attributes, so we can retrieve proper data elements using the attributes. When we retrieve a data element, we use various search options for exact searching. Therefore, various search operations can be used to find a data element. ISO/IEC 11179 has several logical group elements that can group data elements. It includes data element concept, object class, concept domain, etc. This paper defined them as group elements. The group element search is to retrieve them with a given search condition.

The goal of the data element registration is to extend a metadata registry. In other words, in case that there are no data elements, a new data element can be proposed by users. After confirmation, the proposed data element can be registered into the metadata registry. The proposed data element must be filled in with the mandatory attributes according to ISO/IEC 11179. Hence, Data element registration has many detailed operations.

The registration status management of data elements is an operation to change their standardization level. There are six standardization levels: Submitted, Registered, Qualified, Standard, Preferred standard, and Retired. Therefore, we must provide operations for these access patterns.

## 2.2 Metadata Registry Access Patterns

There are many access patterns to create and manage the metadata registries. We can define standardized access interfaces for consistent handling the metadata registries through analysis on the operation patterns. This paper showed the basic operations that are required managing the metadata registries.

In this section, we define and show the detailed operation patterns about the basic operations. This paper focuses the search operations. In the metadata registry management systems, the result of searching is mainly a set of data elements. As aforementioned, a data element has many attributes to depict data. Hence, data elements can be retrieved using the values of the attributes. All of the metadata registries are built according to the standard specification, ISO/IEC 11179. Therefore, they have the same mandatory attributes.

As a result, we can define search operation patterns for retrieving proper data elements. In general, most retrieval systems provide additional search options to retrieve exact results. Thus, the search options can be added the search operations. The analyzed operation patterns of the metadata registry are summarized as follows:

- Search operation of data elements by the mandatory attributes  
: Definition, name, context, etc.
- Data element search operation by data element's registration status  
: Submitted, recorded, qualified, standard, preferred standard, and retired.
- Group element search operations  
: Data element concept, object class, conceptual domain, etc.
- Data element search operations using group elements  
: The group elements are used as qualifiers to retrieve data elements. The final target of these operations is a set of data elements.
- Data element search options  
: Exact matching, partial matching, starting with. These options can be expressed using the original SQL. However, we added these options into the metadata registry operators to be used explicitly.

The group element search operations retrieve data element concepts or object classes. In other hands, the data element search operations using the group elements are to retrieve data elements. Generally, most operations produce a set of data elements as the result of search because the most important object is the data element in the metadata registries. If anonymous metadata registry is built according to the standard specification, the operations above are available to the management systems managing the metadata registries. In other words, all of the standard-based metadata registries produce correct results of the search operation patterns. If we cannot get a result from a metadata registry using the operations, it means that the metadata registry has no instance or has not been built according to the standard. In case of the latter, the metadata registry must be reconstructed or updated to follow the international standard.

### 3 Defining Metadata Registry Query Operators

In this section, we define and describe the metadata registry query operators. These metadata query operators are integrated into SQL3.

#### 3.1 Abstract Query Operators

This paper analyzed and defined the metadata registry query operations for accessing metadata registries. We can define query operators based on the analysis result. We distributed the operators into several conceptual groups, name as abstract query operator. The following table shows the abstract query operators. The basic operations for managing metadata.

**Table 1.** Abstract query operators. These operators are the grouping unit for concrete metadata registry query operators that are integrated into SQL3

Abstract operators	Description
DE_mandatory_attribute_name()	Query operation set for retrieving DEs by mandatory attributes
DE_registration_status()	Query operation set for retrieving DEs by registration statuses
DE_group_element()	Query operation set for retrieving DEs by group elements
group_element()	Query operation set for retrieving group elements

The abstract operators are conceptual operators that generalize concrete query operators. Thus, they can be materialized into concrete query operators. For example, DE\_mandatory\_attribute\_name() is the conceptual operator for concrete query operators that retrieve data elements using their mandatory attributes. Thus, this abstract operator is materialized into concrete query operators such as DE\_name(), DE\_definition(), DE\_context(), and so on. The details of the abstract operators are given in the next section.

These abstract operators are defined to conceptually cluster operations from the analysis of metadata registry access patterns. In other words, they are not actual query operators to be integrated into SQL3. Therefore, they must be realized and materialized into concrete metadata registry query operators. This paper describes it in the next section.

#### 3.2 Concrete Metadata Registry Query Operators

The goal of this paper is to extend SQL and define a standard access interface for consistent access of metadata registries. To achieve this goal, we define metadata registry query operators, concrete query operators based on the operation patterns defined above. The defined query operators are integrated into SQL. Table 2 shows the concrete query operators.

The metadata registry query operators are defined based on the operation patterns already described. The operation patterns, i.e., query operations are provided from the standard specification of ISO/IEC 11179. Therefore, the metadata registry query operators can be used as the standard access interface for handling metadata registries.

Table 2 shows the relationships between the abstract operators and the corresponding concrete query operators for metadata registries. An abstract operator, `DE_mandatory_attribute_name()` is realized into many actual metadata registry query operators. It means that there are many operators by the mandatory attributes specified in ISO/IEC 11179. The examples of the operators are `DE_name()` and `DE_definition()`. `DE_name()` is a operator to retrieve data elements by the name of data elements. The operator `DE_definition()` is to retrieve data elements by the definition of data elements. These concrete query operators may involve KW and/or OPT.

The materialized metadata registry query operators are overloaded. Hence, a same operator can be detailed into several operators with different parameter list. Users can access metadata registries in various manners owing to this overloading. If the operators are included in SQL3, many users who are familiar with SQL3 can easily utilize the metadata registry query operators. This issue will be described in next section.

**Table 2.** Concrete metadata registry query operators. This table shows the concrete query operators for each abstract operator. Each abstract operator is detailed into several query operators that are integrated into SQL3

Abstract operators	Concrete query operators
<code>DE_mandatory_attribute_name()</code>	<code>DE_name(KW, OPT)</code>
	<code>DE_definition((KW, OPT)</code>
	<code>DE_context(KW, OPT)</code>
	<code>DE_reg_ornization(KW, OPT), ...</code>
<code>DE_registration_status()</code>	<code>DE_status(RA, DA, KW, OPT)</code>
	<code>DE_status(RA)</code>
	<code>DE_status_submitted(DA, KW, OPT)</code>
	<code>DE_status_retired(DA, KW, OPT), ...</code>
<code>DE_group_element()</code>	<code>DE_object_class(KW, OPT)</code>
	<code>DE_conceptual_domain(KW, OPT)</code>
	<code>DE_concept(KW, OPT), ...</code>
	<code>object_class(KW, PT)</code>
<code>Group_element()</code>	<code>conceptual_domain(KW, OPT)</code>
	<code>element_concept(KW, OPT), ...</code>

- KW: Keyword given by users
- OPT: Search options (Partial, exact, and starting with)
- RA: Registration attribute
- DA: Mandatory attributes of data elements such as name, definition, context, etc

## 4 Integrating Metadata Registry Operators into SQL3

In this section, the defined operators are integrated into SQL3. This section describes a main syntax of SQL/MDR and several examples.



## 4.1 SQL/MDR Syntax

SQL is familiar to many database users such as researchers, database administrators, application developers and so on. If the metadata registry operators are integrated into SQL3, the existing database users can easily handle the operators.

In this section, we describe the metadata registry query language that is integrated the operators into SQL. The metadata registry query language named SQL/MDR is an extension of the standard query language, SQL. Table 3 shows a brief BNF description for the extended query language SQL/MDR.

**Table 3.** A brief BNF of SQL/MDR. This table shows a partial and main BNF of the whole SQL/MDR syntax

---

```

<extended query specification>
::= SELECT <extended attribute list>
    FROM <extended relation list>
    WHERE <extended attribute qualification>;

<extended attribute list>
::= <attribute list>|<MDR attribute list>|<MDR operator>;

<extended relation list>
::= [COMMA]<general relation list>|<MDR relation list>[<extend attribute list>];

<extended attribute qualification>
::= <general qualification>|<MDR qualification>;

<MDR qualification>
::= [<boolean term>]|<MDR operator>[<extend attribute qualification>];

<MDR operator>
::= <DE mandatory attribute name> | <DE registration status> | DE_STATUS
    L_PAREN<MDR param list>R_PAREN;

<DE mandatory attribute name>
::= NAME | DEFINITION | ... | ORGANIZATION;

<DE registration status>
::= DE_STATUS_SUBMITTED | DE_STATUS_RECORDED | DE_STATUS_QUALIFIED
    | DE_STATUS_STANDARD | DE_STATUS_PREFERRED | DE_STATUS_RETIRED;

<MDR relation list>
::= DATA_ELEMENT | DATA_ELEMENT_CONCEPT | ... | OBJECT_CLASS;

```

---

SQL/MDR is an extension of SQL3 to provide a consistent access interface for metadata registries. Before extending SQL3, we first defined a standardized interface of table names and their attribute names used in metadata registries. The interface means a set of predefined and promised table names and attribute names that can be validly used in all metadata registries according to ISO/IEC 11179. Therefore, the interface allows metadata registry query statements to be simplified.

Most of all, we can verify the built metadata registries are valid or not. In addition, it improves interoperability between metadata registries that have been built independently and remotely.

## 4.2 Examples

This section describes several examples of metadata registry query. We first show the query statement to directly access a database using only SQL3, and show the query statement using the extended query language.

**Query 1.** Retrieve all data elements where registration status is 'RECORDED'.

```
SQL/MDR> SELECT DE_status (RECORDED)
          FROM data_element (Q1-a)
```

This query statement (Q1-a) is written using a metadata query operator DE\_status() of the extended query language SQL/MDR. In case of SQL, we must write WHERE clause. As a result, SQL/MDR allows the same query statement to be written simply. It also derives to be built following the international standard ISO/IEC 11179.

**Query 2.** List all data elements where their registration status is 'RECORDED' and name is matching to 'JOURNAL'.

```
SQL/MDR> SELECT * FROM data_element
          WHERE DE_status (RECORDED)
          AND name= '%JOURNAL%' (Q2-a)
```

```
SQL/MDR> SELECT * FROM data_element
          WHERE DE_status (RECORDED, name, 'JOURNAL') (Q2-b)
```

```
SQL/MDR> SELECT * FROM data_element
          WHERE DE_status_recorded (name, 'JOURNAL') (Q2-c)
```

Query 2 can be written in various manners using the metadata query operators. Three query statements for Query 2 are shown above. The first, (Q2-a) is an example written with the metadata registry operator and the conventional query statement. The queries, (Q2-b) and (Q2-c) are similar but have different parameters. Users can select and write their query statements with proper operators.

### 4.3 Evaluation and Discussion

SQL/MDR is the query language allowing the standardized and consistent access method for the metadata registries. It is based on and extended the standard query language, SQL3, which is used broadly. Therefore, many database users can use easily it for access metadata registries. Also the mechanism for sharing and exchanging between metadata registries is not complex owing to its standardized access method.

The previous metadata registry management systems have different access methods respectively. It causes many problems. First, high cost is required to achieve interoperability between them. Also, the mechanism of exchanging and sharing is very complicated. Many query statements for one request should be composed and processed.

Query 3 is described to show SQL/MDR efficiency. Query 3 is an example to access several distributed metadata registries and its request is the same with Query 1. We first assume the following situation to show several contributions of SQL/MDR.

#### *Assumption:*

- There exist two metadata registries.
- Each metadata registry has different MDR structure each other.
- The first metadata registry, MDR1 is designed with the following structure.
  - data\_element\_table includes all of the data elements
  - data\_element\_name is a field name of data\_element\_table and means name of data elements

- Status is a field name of data\_element\_table and means registration status of data elements
- The second metadata registry, MDR2 is designed with the following structure.
  - table1 and table2 include all of the data elements together. Most attributes are included in table1 and some attributes including registration status are in table2.
  - table1 and table use name as join key.
  - Registration status is involved in reg\_status, a field of table2.

**Query 3.** (Distributed access to two metadata registries) Retrieve name of all data elements where registration status is 'RECORDED' from metadata registries, MDR1 and MDR2.

*SQL/MDR*> SELECT DE\_status (RECORDED)  
FROM data\_element (Q3-a)

*SQL<sub>MDR1</sub>*> SELECT data\_element\_name  
FROM data\_element\_table  
WHERE status = 'RECORDED' (Q3-b)

*SQL<sub>MDR2</sub>*> SELECT table1.name  
FROM table1, table2  
WHERE table1.name=table2.name  
AND table2.reg\_status = 'RECORDED' (Q3-c)

If two metadata registries are designed and developed based on SQL/MDR approach, only one query statement (Q3-a) is written and processed to get the final result. However, the previous approach requires two query statements, (Q3-b) and (Q3-c) because they have different metadata registry structure and also there is no consistent access interface between them. If the number of metadata registries be  $N$ , then the previous approach requires  $N$ -query statements. Therefore, query modeling cost, distributed query process cost, preprocessing cost, and complexity of exchanging mechanism increase exponentially. Table 4 shows comparisons of SQL/MDR approach and previous approach.

**Table 4.** Qualitative comparison. The comparative items in this table can be divided into two aspects (Query modeling aspect and system development and query processing aspect)

Comparative item	Previous approach	SQL/MDR
Simplicity of query statement	Complex	Simple
Ease to use	Different & unfamiliar	Easy (SQL-like)
Familiarity with users	Low (Various)	High (SQL-like)
Independency of query description	Dependent on each system	Independent (Uniform method)
Query modeling cost	High (query rewriting)	Low (neutral modeling)
Module for interoperability between MDRs	N/A	Support
Distributed query processing cost	High (n-to-n adapter)	Low
Standardization of access method	N/A	Support
Complexity of exchanging and sharing mechanism	High	Low

In addition, SQL/MDR is an extension of SQL3, which is the international standard for query language of databases. Most of the database handlers are familiar to SQL3. Thus, they can utilize SQL/MDR in easy.

## 5 Conclusion

The metadata registry was inherited from the international standard ISO/IEC 11179. Currently, many metadata registries have been built in various fields. Although the metadata registries follow the standard and there are the same operations to access them, there is no standard method to consistently access the metadata registries.

In this paper, we supposed a metadata registry query language as an access method to consistently manage the metadata registries in a standard manner. The metadata registry query language is named as SQL/MDR and is an extension of the international standard query language, SQL3. To achieve this goal, we analyzed basic operations that are required to access the metadata registries. Then we defined the metadata registry operators based on the analysis result. Finally, these metadata registry query operators were integrated into SQL3.

The proposed SQL/MDR is an extension of SQL3. Thus, many users managing databases can easily utilize SQL/MDR. SQL/MDR provides the standard names for essential composing elements that should be defined to create a metadata registry. In other words, we can identify whether anonymous metadata registry follows ISO/IEC 11179 specification or not. It encourages all metadata registries to be created according to the standard specification. In addition, SQL/MDR can be used as a protocol to exchange and share data elements between distributed metadata registries built independently.

In future, additional metadata query operators must be defined to increase usability of SQL/MDR. This paper focused on defining only query language for searching. Therefore, various query operators for creation, delete, and update should be defined and integrated into the proposed query language.

## References

1. ISO/IEC JTC 1/SC 32, ISO/IEC 11179: Specification and standardization of data elements Part 1~6. ISO/IEC JTC 1 (2003)
2. Korea Institute of Science and Technology Information, A study on the development of standardization and management model for science and technology information. Research Report (2002)
3. Electronics and Telecommunication Research Institute, Research on the Registration and Search System of Component, Research Report (2000)
4. Environmental Protection Agency, Environmental Data Registry, <http://www.epa.gov/edr/>. (2004)
5. Australian Institute of Health and Welfare, Australian National Health Information Knowledgebase, <http://www.aihw.gov.au/> (2003)

6. U.S. Transportation System, <http://www.dot.gov/>. U.S. Department of Transportation (2003)
7. Egenhofer, M., Spatial SQL: A query and presentation language. IEEE Transactions on Knowledge and Data Engineering (1994)
8. Pissinou, N., Snodgrass, R., et al., Towards an infrastructure for temporal databases: Report of an invitational arpa/nsf workshop. In SIGMOD Record (1994)
9. ISO/IEC JTC 1/SC 32, ISO/IEC 13249: Information technology - Database languages - SQL Multimedia and Application Packages, ISO/IEC JTC 1 (2003)
10. Environmental Protection Agency, Data Standards Publications and Guidances (2003)
11. Australian National Health Data Committee, National Health Data Dictionary (2003)
12. ITS Architecture Development Team, ITS Logical Architecture- Volume I, Volume II: Process Specifications. Volume III: Data Dictionary (2003)
13. ISO/IEC JTC 1/SC 32, ISO/IEC 9075: Database Language SQL3 Part 1~10. ISO/IEC JTC 1 (1999)
14. Lee, J.-Y., Integrating Spatial and Temporal Relationship Operators into SQL3 for Historical Data Management. Journal of Electronics and Telecommunication Research Institute, Vol. 24, No. 3. Electronics and Tele-communications Research Institute (2002) 226-238
15. Jeong, D., Park, S.-Y., Baik, D.-K, A Practical Approach: Localization-based Global Metadata Registry for Progressive Data Integration. Journal of Information & Knowledge Management, Vol. 2, No. 4. World Scientific (2003) 391-402

# Solving Queries over Semantically Integrated Biological Data Sources

José Francisco Aldana-Montes, Ismael Navas-Delgado,  
and María del Mar Roldán-García

University of Málaga, Computer Languages and Computing Science Department  
Higher Technical School of Computer Science Engineering, Spain,  
`{jfam,ismael,mmar}@lcc.uma.es`  
<http://khaos.uma.es/>

**Abstract.** Mediators used to be developed as monolithic systems that envelop the data source's information, such as its semantics and location. It involves a high degree of coupling among the mediator's components. This coupling does not allow sharing services with other organizations or the dynamic integration of new data sources. We propose an architecture for conceptual mediation in which the sources' query capabilities are published as web data services. This architecture also provides a way to achieve interoperability between applications in the Semantic Web context.

## 1 Introduction

The amount of information in the web and the complexity of processing it in a reason-able manner has given rise to a lot of research on heterogeneous data integration. Mediator-Wrapper is the most common approach to achieve this goal. Its main objective is to allow users to make complex queries over heterogeneous data sources, as if they were a single one, using an integration schema. Mediators offer users interfaces for querying the system, based on the integration schema. These interfaces transform user queries into a set of sub-queries that are sent to the wrappers in order to be translated to the data sources schema. Therefore, data sources can solve each sub-query. Usually, sub-query results are unstructured documents that are translated into structured documents following a standard format (XML, for example). XML technology makes it possible both to structure the information and to explicit the schema by means of an XMLSchema document. However, this information is insufficient for agents searching the Web, due to the fact that they cannot interpret these XML documents because they do not know their semantics.

Mediation systems have evolved by means of different improvements made to traditional mediation architecture (of systems such as Manifold [1] and TSIMMIS [2]). Nonetheless, there are still several open problems in semantic heterogeneous data integration:

1. Design problems: (1) mediators and wrappers are strongly coupled; (2) they are manually designed for each new mediator by software developers; in other

words, there is no reusability; (3) there is no dynamic integration; (4) it is not possible to use loose integration; and (5) software agents can not find wrappers, as they are “hidden” behind mediators.

2. Semantic problems: (1) wrapper query capabilities and semantics are not published; and (2) traditional mediators do not allow expressive queries based on semantics.

On the other hand, the accumulated biological knowledge needed to produce a more complete view of any biological process is disseminated around the world in form of biological sequences, structures, protein functions and pathway databases. It is frequently stated that the volume of data in molecular biology is growing at exponential rates. Nonetheless, the key feature of biological data is not so much its volume but rather its diversity, heterogeneity and dispersion [3]. Thus, we believe that mediation is a good approach for integrating biological information sources. For this reason we have also studied several biological mediators, such as TAMBIS [4], BioDataServer [5], KIND [6] and BioKleisli [7]. And we have developed an XML mediator, namely BioBroker [8], which reveals the need of a way to dynamically integrate data sources in the field of biology. The need of systems that provide dynamic integration of data sources has led us to define a novel semantic mediation architecture [9] (Section 2). This architecture includes directories in which an ontology and several resources with semantics relevant to the domain information are published. We also improve the wrapper generation process by publishing them as web services and making their semantics accessible. This evolution from traditional wrappers to data services is motivated by the following goals:

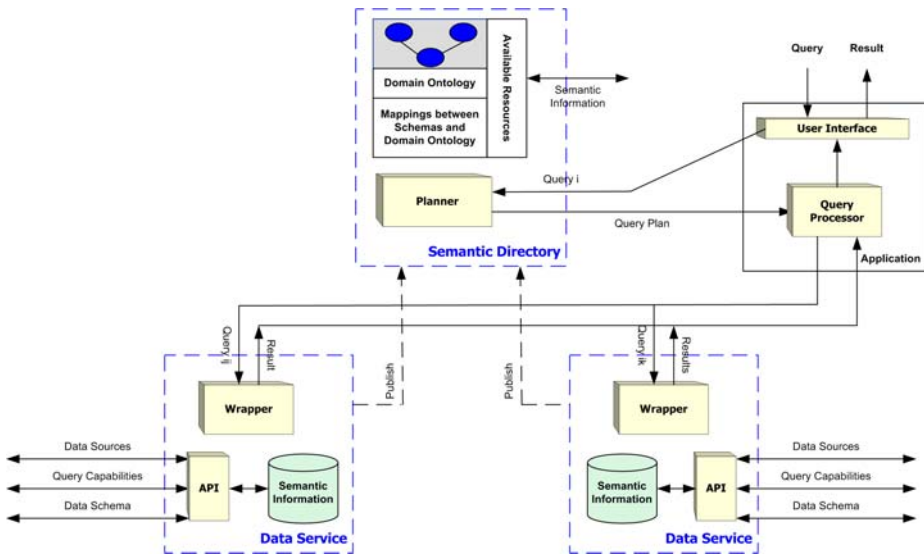
- The data services can be used by other mediators.
- The semantics of the data services is published on the web to be available for other applications.
- Wrapper query capabilities can be enveloped into one or more services.

In this paper we briefly describe our architecture (Section 2). In Section 2.1 we present an extension of our proposal. Then, we illustrate by means of an example, how our system can solve queries that other mediators cannot solve. Finally, Section 3 presents some conclusions and future work.

## 2 DCRONO: A Semantic Mediation Architecture

DCRONO (Dynamic Conceptual integRation of heterOgeNeous data sOurces) is a framework architecture that seeks to make wrappers independent entities and to eliminate their ties to the mediator, thus increasing their reusability in different applications. We emulate P2P hybrid systems, which implement a directory with location information of available resources. In these systems the applications access the resources directly by means of point to point connections that the directory has provided them with. Therefore, the flow of information

is greatly reduced w.r.t. the one generated in the traditional client-server architectures. Our proposal for semantic mediation stems from the need for both dynamic integration and applications interoperability. This architecture introduces two main considerations concerning the basic architecture of mediation: (1) the isolation of wrappers, which are encapsulated as web services (Data Services for us); and (2) the added directory (Semantic Directory) with information about these Data Services (See Figure 1). This architecture allows wrappers to contribute data, information schemas and query capabilities in a decentralized and easily extensible way.



**Fig. 1.** Architecture for Semantic Integration

A data service needs to be registered in one or more semantic directories in order to be used by a mediator or other software agent. In other words, data services, like P2P hybrid systems, must know the location of semantic directories that are in the same application domain. Finally, public interfaces of data services and semantic directories will allow applications that share their communication protocol to take advantage of knowledge about available directory resources. Next we present the components of the proposed architecture as well as their functionality.

Semantic directories offer essential services for query processing, and data services provide the minimal elements for solving query plans. The aim of this type of component is to allow applications to access wrapper functionalities. In this way, we have designed an extensible and adaptive architecture in which we can define a data service as “a service that offers wrapper query capabilities using web protocols”.



The publication of these online web services using UDDI (Universal Description Discovery Integration) [10] could allow other applications to dynamically discover wrappers by means of an easy interface. However, our data services have been devised for publication in a specialized type of directory: the semantic directory.

Semantic directories are at the core of this architecture because they provide essential services for solving user queries. We can define a semantic directory as “a server that offers information about available web resources (data services), a domain ontology, mappings between resource schemas and this ontology, and provides a query planner”.

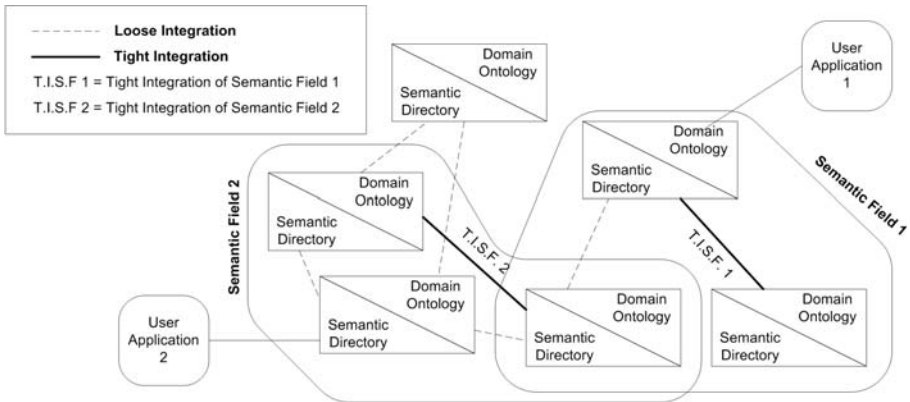
A semantic directory stores an ontology described with OWL [11]. The planner can use reasoning mechanisms over this ontology in order to obtain better query plans. This ontology, which must be generic for the application domain, describes the core knowledge that is shared by a set of applications. It can be seen as an abstraction of the knowledge of the resource schemas. Each schema could be considered as a refinement of the domain ontology. Information about data services will be added to a semantic directory when services register in it.

## 2.1 Semantic Fields: The Use of DCRONO on the Semantic Web

The basic configuration of our proposal has been described with only one directory. But we will usually have distributed configurations and/or more complex architectures with several semantic directories. Now we present an analysis of the integration problem between semantic directories in order to obtain interoperability capabilities among several application domains. We propose a two level integration solution. At a first level we use “loose” integration between close domains. For this task we take advantage of a reference ontology and use automatic matching algorithms based on semantic distance measurements. At the second level we propose “tight” integration among non nearly ontologies (see Figure 2).

**Semantic Field:** Semantic directories integrate data sources taking advantage of domain ontologies, as shown in Section 2. These ontologies represent core knowledge that is shared by a set of resources. However, these domains will not be sufficient for several end-users, who could need to obtain information from different, but related, domains.

For example, a semantic directory can include a molecular ontology, but users may need to retrieve information about molecular and structural data. In this case, this application can use “loose” mappings with other related-domain ontologies in order to solve user queries. We define “loose mappings” as “mappings that can be automatically obtained without human interaction”. Ontologies related by means of loose mappings set up a semantic field, which is dynamically built up and updated, for a given user-application perspective. Thus, we provide easy and automatic integration to offer users access to a set of resources, apart from the resources of its semantic directory.



**Fig. 2.** Mediation on the Semantic Web

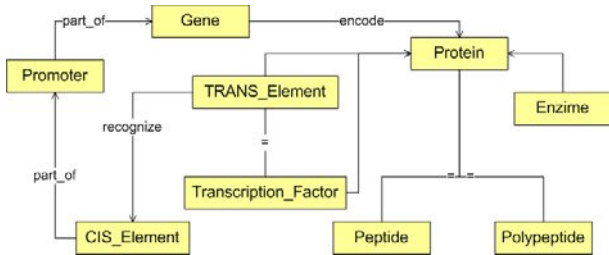
An application builds a semantic field selecting a semantic directory and starting to search the nearest ontologies of other semantic directories (setting a maximum level of remoteness to compare with calculated semantic distances). Once the application establishes a set of semantic directories it can use, it will have defined a semantic field centered on a reference ontology (the one corresponding to the initial semantics). Next, end-user queries are decomposed to be planned for the semantic directories of the semantic field. Then these plans are combined and evaluated to return a complete result to the user.

Semantic fields are generated, evolve dynamically and depend on the application's perspective, but applications need to find semantic directories to build semantic fields. For this task the system takes advantage of a P2P network. That is, when a new semantic directory appears it is published in a P2P network, each semantic directory and application of this network receives information about the new directory, and the semantic distance to other semantic directories of the network are calculated. Note that semantic distances are static and only change when an ontology changes.

However, applications may need to use a domain ontology that has not been automatically added to its semantic field. In this case, the application can generate "tight" mappings between ontologies of its semantic field and external ontologies. But institutions with standardization capability can also define these kinds of mappings between well-known domain ontologies. This last type of tight mappings is static and does not depend on the application perspective. We define "tight mappings" as "those mappings that cannot be automatically obtained and are established with human inter-action". When semantic directories are mapped with a domain ontology of a semantic field, they are added to the semantic field. Note that this type of integration could involve new loose mappings with ontologies from outside our semantic field. That is, if we add a new directory to our semantic field, then new loose relationships are searched in order to include new directories in it.

Our proposal is not to obtain isolated semantic fields; instead, we hope that our solution will offer capabilities to achieve interoperable applications. In consequence, we propose using both loose and tight mappings to obtain semantic fields, and to take advantage of them to solve queries.

Suppose that we have a semantic directory with a domain ontology about biological functions and another one that builds a structural ontology like a domain ontology. When our application (that works with the first directory) receives information about the second directory, taking advantage of loose matching algorithms it includes this new directory in its semantic field. Now this application may take advantage of the new semantic directory to solve its user queries, as for example: “Find structural information about Transcription Factors, which are proteins that recognize CIS elements of promoters of genes that encode the enzyme aldehyde dehydrogenase”. Thus, the functional part of the query is solved in the directory that has the functional ontology, and the structural information is retrieved from resources of the other directory. Next, another semantic directory is published with a biological digital library ontology, and we establish a tight mapping between its ontology and this new ontology. Now it can solve more complex queries such as: “Find structural information about Transcription Factors, which are proteins that regulate CIS elements of promoters of genes that encode the enzyme aldehyde dehydrogenase, and publications where information about these regulations can be found”. This query includes references to the three directories, so it is necessary to send each part to the relevant semantic directory.



**Fig. 3.** Biological Ontology

## 2.2 Biological Use Case

We present an example of use of the described proposal for the integration of biological data sources. As a first step we have generated a domain ontology to be used in a semantic directory. This ontology represents a functional view of biological data (see Figure 3). Once the directory has been developed, we can implement applications that will use it. These applications must include an evaluator and maybe user interfaces.

Once semantic directories have been developed, they are autonomous and do not need human actions, but a semantic directory and an application are not

enough to solve user queries. So we have developed data services and added them to the semantic directory (taking advantage of the “connect” method). These resources envelop the access to biological sources, such as SRS LOCUSLINK, EMBL, SWISS-PROT, SWISS-PROT ENZYME, PDB, MICADO, DIP, EPD and TRANSFAC. Mappings between resource schemata and the domain ontology are now obtained manually and inserted in the semantic directory when connected to a data service.

Now we can solve a query like “Find the nucleotides sequence of proteins encoded by some gene that contains the promoter whose id is EP17030”. This query is rewritten in logical terms:

`ans(S) :- Protein(P), Gene(G), encode(G,P), sequence(P,S), Promoter(Po), part_of(Po,G), id(Po, EP17030)`

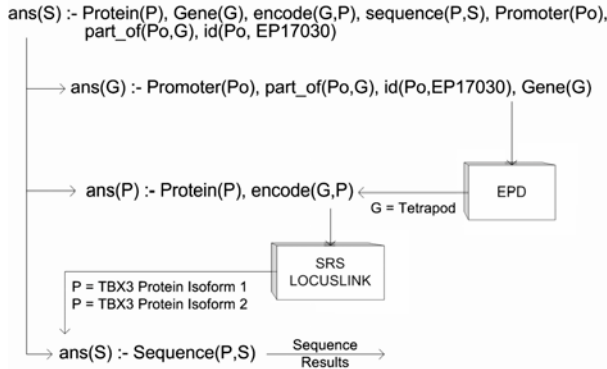
This query is sent to the semantic directory, which analyzes it taking advantage of mappings between resources and the domain ontology. The planner takes into account query capabilities in order to find sub-queries for each data source. Finally, the query plan is evaluated by the user application, querying each data source and composing results in terms of the domain ontology. Using all these mappings, the proposed query is divided into three sub-queries:

1. `ans(G) :- Promoter(Po), part_of(Po,G), id(Po,EP17030), Gene(G)`
2. `ans(P) :- Protein(P), encode(G,P)`
3. `ans(S) :- Sequence(P,S)`

These queries are connected by means of the gene (G) and the protein (P). So the evaluation must be sequential (in a non-optimized plan). Now, these queries must be evaluated in resources in which they can be solved. For this task the semantic directory also makes use of mappings. Thus, it is sent to these resources, and we obtain the results of Figure 4. Once the gene has been obtained, we can try to solve the second sub-query to obtain the protein. Finally, the nucleotide sequences are obtained and the system achieves the user-query result. If we perform this task with a traditional mediator (that uses an ontology as the integration schema), we can only send the second query to the SRS LOCUSLINK data service. However, using our architecture, before solving these queries we can use inference mechanisms to get better query plans. Thus, we use the class-subclass inference mechanism to determine that instances of TRANS\_Element and Enzyme classes are also instances of Protein class. Using this knowledge our architecture sends the second sub-query to the TRANSFAC and Swissprot ENZYME data services. In this way, we can obtain more results than if we only use the first service, which is what happens in traditional mediators. This sub-query is rewritten as:

1. `ans(P) :- Protein(P), encode(G,P)`
2. `ans(P) :- TRANS_Element(P), encode(G,P)`
3. `ans(P) :- Enzyme(P), encode(G,P)`

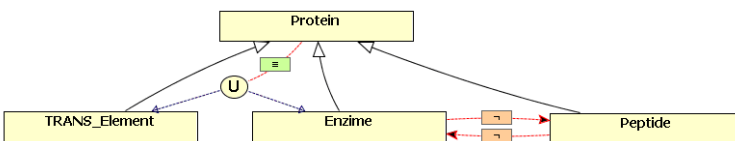
Finally, sub-query results are composed in order to obtain ontology instances that will be returned to the end-user. In our example these instances include received data, but the system removes duplicate instances.



**Fig. 4.** Example of resolution

**Advanced Queries:** This section tries to illustrate the differences between traditional mediators and the proposed architecture with respect to query complexity. By exploiting the reasoning capabilities of ontologies it is possible to solve queries that are impossible to solve by means of traditional systems. These reasoning capabilities allow us to derive new knowledge that the ontology does not explicitly describe. Ontology definition languages like OWL provide the possibility of adding rich semantic information to the ontologies; this information can be used to infer knowledge that helps us improve our queries. For example, the queries of the previous example can also use the equivalent class knowledge to infer equivalent classes of Protein and TRANS\_Element. Thus, queries that include these classes can be rewritten to improve queries with this new knowledge:

1. ans(P) :- Protein(P), encode(G,P) a. ans(P) :- Polypeptide(P), encode(G,P)  
b. ans(P) :- Peptide(P), encode(G,P)
2. ans(P) :- TRANS\_Element(P), encode(G,P) a. ans(P) :- Transcription\_Factor(P), encode(G,P)



**Fig. 5.** Complex Query (Disjoint Classes)

In order to clarify these ideas we present some extensions of the previous ontology and we describe how we can use the ontology knowledge to derive new knowledge. Suppose that we add to the ontology some knowledge about which classes are disjoint classes. For example, an enzyme cannot be a peptide, thus Enzyme and Peptide are disjoint classes. (Figure 5). Since a protein has to be

a TRANS\_Element or an Enzyme and always one of them, we can infer that every Peptide is also a TRANS\_Element. Therefore, a query such as “Find all trans elements whose nucleotide sequence contains MAYHPFH” will return all trans elements that has the sequence MAYHPFH plus the peptides that commit with this assertion (because we infer that a peptide is also a trans element). A traditional mediator has no mechanism to identify this information.

Finally, the ontology definition language allows us to define special properties of relationships. These properties also encapsulate knowledge about the domain that it is not explicitly asserted. For example, we can define a relationship as a transitive one. We extend our ontology defining that the relationship `part_of` is a transitive relationship. Therefore, we could assert that a Promoter is part of a Gene and a CIS Element is part of a Promoter. Since the relationship “`part_of`” is transitive, we can infer that a CIS Element is part of a Protein. This information can also be used to make a query like “find all CIS Elements that are part of a Gene named `atpA`”. In a traditional mediator it is not possible to know that a Gene has CIS Elements, because this information will not be in the integration schema.

### 3 Conclusions and Future Work

Our proposal makes use of existing improvements and uses them in an integration architecture. In this way, we can handle new problems over an architecture that takes advantage of other solutions. Thus, we explicitly describe mediator and wrapper semantics. We study design and use existent technology (web services, P2P, etc.) to obtain better integration systems.

In this paper we present an architecture (DCRONO) to integrate data sources and an example for integrating Biological sources. This architecture is based on an extension of traditional mediation, called semantic mediation. A mediator typically uses an integration schema as a global view of the local schemas of the data sources that are integrated. In our approach, the semantics introduced in the Semantic Directories allows users to make more expressive queries, namely semantic queries. Furthermore, information inferred from the ontology-explicit knowledge is used to make queries that a traditional mediator could not evaluate.

We provide elements to achieve interoperability between semantic integration systems that cooperate in the same application domain or have certain relations (Semantic Fields). Furthermore, we introduce a solution to integrate semantic fields and obtain better query capabilities. This will allow us to make complex queries, relating do-mains like Digital Libraries and Molecular Biology. These kinds of queries will really exploit the advantages of the proposed architecture.

Therefore, in several domains in which there are no technical users, such as librarians or biologists, dynamic integration is a very important issue. In this context, it is necessary to give users a simple environment for integrating data information without modifying the mediator code or schema. Directories supply an easy way to integrate data sources, opening up new directions for dynamic integration.

As future work, we intend to study the possibility of giving data services more semantics, taking into account service quality, relations with other domain ontologies, etc. We plan to continue studying automatic mapping between schemas and ontologies to establish correspondences between a retrieved document's schemas and directory ontologies in those new systems developed using our architecture. We are also interested in establishing a semantic model to define the data service's query capabilities, which improves query planning by adding inferences about query capabilities to reasoning between schemas and the domain ontology.

## Acknowledgements

This work has been supported by the MCyT grant (TIC2002-04186-C04-04).

## References

1. Levy, A., Rajaraman, A., Ordille, J.: Querying Heterogeneous Information Sources Using Source Descriptions. In Proc. VLDB, pages 251-262, Mumbai (Bombay), India, Sep-tember 1996.
2. Papakonstantinou, Y., Garcia-Molina, H., Widom, J.: Object Exchange Across Heterogeneous Information Sources. In Proc. ICDE, pages 251-260, Taipei, Taiwan, March 1995.
3. Rechenmann, F.: From Data to Knowledge. *Bioinformatics*, v. 16, n. 5 pp 411.
4. Stevens et al.: TAMBIS: Transparent Access to Multiple Bioinformatics Information Sources. *Bioinformatics*, 16:2 PP. 184-186.
5. Lange et al.: A computational Support for Access to Integrated Molecular Biology Data.
6. Gupta, A., Ludäscher, B., Martone, M.: Knowledge-based Integration of Neuroscience Data Sources. 12th Int. Conf. Scientific and Statistical Database Management Systems, Berlin, 39-52, 2000.
7. Davidson, S. et al.: BioKleisli: A Digital Library for Biomedical Researchers. *International Journal of Digital Libraries*, 1(1):36-53, April 1997.
8. Aldana et al.: Integrating Biological Data Sources and Data Analysis Tools through Mediators. SAC 2004.
9. Aldana, J.F., Navas, I.: Towards Conceptual Mediation. ICEIS 2004.
10. UDDI Spec Technical Committee Specification. <http://uddi.org/pubs/uddi-v3.0.1-20031014.pdf>.
11. OWL Web Ontology Language 1.0 Reference. <http://www.w3.org/TR/owl-ref/>

# Fast Approximate Search in Text Databases

Fei Shi

Computer Science Department, Suffolk University  
Beacon Hill, Boston, MA 02114, USA  
`shi@mcs.suffolk.edu`

**Abstract.** We present an index structure that, for a given keyword  $T$  and a tolerance value  $k$ , can find all documents in the database that contain the keyword  $T$  or other words that match  $T$  approximately (We say that two words match each other approximately if the *edit distance* between them does not exceed the tolerance value  $k$ ). Let  $M$  denote the sum of the lengths of all words in the vocabulary and  $z$  the number of words in the vocabulary. The index takes  $O(M \log z)$  time to construct and occupies  $O(M)$  storage. The locations of all the documents in the database that contain the keyword  $T$  or words that match  $T$  approximately, can be found in  $O(n(\log z + o n))$  expected time where  $n$  is the length of the keyword  $T$ ,  $z$  the number of words in the vocabulary, and  $o$  the number of words in the vocabulary that match  $T$  with an edit distance of at most  $k$ .

## 1 Introduction

The retrieval of documents that match given keywords is one of the main problems in document database systems. This problem is gaining greater importance recently, largely due to the use of the World-Wide Web and the feasibility of keeping all documents on-line.

In this paper, we propose an index structure to support the *approximate keyword search*. In an approximate keyword search query, the user presents a keyword  $T$  (called the query word) and a tolerance value  $k$  ( $k \geq 0$ ), and wishes to find all the documents in the database that contain the keyword  $T$  or words that match  $T$  with at most  $k$  errors (differences).

We use the *edit distance* to measure the difference between two words. The edit distance between two words (strings) is the minimal number of edit operations (insertions, deletions, or changes of characters) that must be performed to convert one word into the other.

The approximate search is useful in many applications. For example, a user may misspell a word due to human error. The same message transmitted repeatedly on the same communication channel may be received differently on different occasions due to the noise on the channel. In molecular biology, one often needs to compare a newly sequenced protein against a (huge) dictionary of protein sequences to see if it matches, allowing certain number of errors, any protein already stored in the dictionary. Hereby, the exact matching paradigm



is inappropriate since even sequences of the same protein differ slightly among organisms of the same species.

As far as we know, most existing document database systems that can perform approximate keyword search must first scan the vocabulary sequentially, word by word. Once the set of matching words in the vocabulary is found, their documents in the database can be retrieved using, for example, inverted lists. The index structure to be presented in this paper does not search the vocabulary sequentially, it instead uses a filtering technique that allows us to examine only a small set of words in the vocabulary that are likely to match the query, thus reducing search time significantly.

## 2 Computing Edit Distances between Strings

Several distance functions have been proposed to measure the difference between strings. The edit distance is one of the major difference measures for string processing. It has been used in many areas, such as text databases, molecular biology, and data communications.

**Definition 1 (edit distance).** *The edit distance between two strings  $X$  and  $Y$ , denoted as  $D_e(X, Y)$ , is defined as the minimal number of edit operations - deletions, insertions, and substitutions of characters - needed to transform one string into the other.*

A simple dynamic programming algorithm to compute the edit distance has been discovered independently by several authors [SK83]. The algorithm runs in  $\Theta(mn)$  time. In fact,  $\Omega(mn)$  operations are necessary to compute the distance if operations on characters of the strings are restricted to tests of equality [WC76].

## 3 Computing $q$ -Gram Distances

Since computing the edit distance requires quadratic time, we will use a “simpler” distance, called the  $q$ -gram distance to estimate the edit distance. The  $q$ -gram distance between two strings can be computed in linear time. Let  $\Sigma$  be a finite alphabet. Let  $\Sigma^*$  denote the set of all strings over  $\Sigma$  and  $\Sigma^q$  denote the set of all strings of length  $q$  over  $\Sigma$ , sorted in lexicographic order, for some integer  $q$ . A  $q$ -gram is any string  $w = x_1 \cdots x_q$  in  $\Sigma^q$ .

**Definition 2 ( $q$ -gram profile/distance).** *The  $q$ -gram profile  $G(S)$  of a string  $S \in \Sigma^*$  is a vector of  $|\Sigma|^q$  dimensions whose  $i$ -th element equals the number of occurrences of the  $i$ -th  $q$ -gram of  $\Sigma^q$  in  $S$ . The  $q$ -gram distance  $D_q(S_1, S_2)$  between two strings  $S_1$  and  $S_2$  in  $\Sigma^*$  is defined to be the  $L_1$ -distance of their  $q$ -gram profiles.*

*Example 1.* Let  $S_1 = 001111$  and  $S_2 = 01010$  be two strings in the binary alphabet  $\Sigma = \{0, 1\}$ . Their 2-gram profiles, listed in the lexicographical order of the 2-grams, are  $G(S_1) = (1, 1, 0, 3)$  and  $G(S_2) = (0, 2, 2, 0)$ , and their 2-gram distance is  $D_2(S_1, S_2) = 7$ .

The concept of  $q$ -grams dates back to Shannon [Sha48] and has been applied successfully in many variations in different string processing methods (see, e.g., [OM88] and [Ukk92]).

### 3.1 Constructing $q$ -Gram Profiles

Let the alphabet be  $\Sigma = \{a_0, a_1, \dots, a_{z-1}\}$  with  $z = |\Sigma|$ . Any string  $X = x_1x_2 \dots x_q$  can be interpreted as an integer  $f(X)$  in base- $|\Sigma|$ -notation:

$$f(X) = \sum_{i=1}^q \bar{x}_i z^{q-i} \quad (1)$$

where  $\bar{x}_i = j$  if  $x_i = a_j$ ,  $i = 1, 2, \dots, q$  (that is,  $x_i$  is the  $j$ -th character in the alphabet  $\Sigma$ ). Note that  $f(X)$  is a bijection, i.e.,  $f(X) = f(X')$  if and only if  $X = X'$ . We call  $f(X)$  the finger-print of  $X$ . When “scanning” a string  $T = t_1t_2 \dots t_n \in \Sigma^*$ , i.e, when considering  $q$ -grams in adjacent positions, the finger-prints can be computed quickly. More precisely, for  $W_i = t_it_{i+1} \dots t_{i+q-1}$ ,  $1 \leq i \leq n - q + 1$ , we have

$$f(W_{i+1}) = (f(W_i) - \bar{t}_i z^{q-1})z + \bar{t}_{i+q} \quad (2)$$

By first computing  $f(W_1) = \sum_{i=1}^q \bar{t}_i z^{q-i}$  and then applying Eq. (2) for  $i = 1, 2, \dots, n - q$ , all finger-prints of all  $q$ -grams in  $T$  can be computed in  $O(n)$  time. Each time when the finger-print of a  $q$ -gram  $W_i$  is computed, we record its occurrence in  $T$  in an array  $G(T)[0 : z^q - 1]$  by incrementing  $G(T)[f(W_i)]$  by 1. So, we get the following result:

**Lemma 1.** *The  $q$ -gram profile  $G(T)$  of a string  $T$  of length  $n$  can be computed in  $O(n)$  time and occupies  $O(|\Sigma|^q)$  space.*

Since at most  $n - q + 1$  elements of the  $q$ -gram profile  $G(T)[0 : |\Sigma|^q - 1]$  of  $T$  are active (non-zero), the space requirement of  $O(|\Sigma|^q)$  can be reduced to  $O(n)$  with some standard hashing technique on the non-zero elements of  $G(T)[0 : |\Sigma|^q - 1]$ , resulting in a compact version of the  $q$ -gram profile  $CG(T)$  of  $T$ . Namely, the compact  $q$ -gram profile  $CG(T)$  of  $T$  is a table of  $n - q + 1$  entries. For each distinct  $q$ -gram  $X$  of  $T$ , there is an entry  $\langle f, c \rangle$  in the table where  $f$  equals the finger-print of  $X$ , and  $c$  is the number of times  $X$  appears in  $T$ . It is easy to see the following lemma.

**Lemma 2.** *The compact  $q$ -gram profile  $CG(T)$  of a string  $T$  of length  $n$  can be computed in  $O(n)$  expected time and occupies  $O(n)$  space.*

### 3.2 Computing $q$ -Gram Distances

From Lemma 1, it is easy to see

**Lemma 3.** *Given the  $q$ -gram profiles of two strings  $T$  and  $P$  of length  $n$  and  $m$  respectively, the  $q$ -gram distance  $D_q(T, P)$  between  $T$  and  $P$  can be computed in  $O(n + m)$  time.*

### 3.3 Using $q$ -Gram Distance to Estimate Edit Distance

Since any algorithm that computes the edit distance between two strings  $X$  of length  $n$  and  $Y$  of length  $m$  needs to spend  $\Omega(nm)$  time (see, e.g., [WC76] and [AHU76]), we would like to estimate the edit distance by  $q$ -gram distance which can be computed in linear time.

The following fact suggests that  $q$ -gram distance can be used as a lower bound for the edit distance.

**Lemma 4 (Ukkonen[Ukk92]).** *Let  $P = p_1 p_2 \cdots p_m$  and  $T = t_1 t_2 \cdots t_n$  ( $n \geq m$ ) be any two strings and  $q$  any positive integer,  $q < m$ . Then,*

$$D_q(P, T)/2q \leq D_{ed}(P, T).$$

The  $q$ -gram distance measure has been successfully used by a number of researchers as a filter to speed up algorithms for the approximate string matching problem (see, e.g., [Ukk92] and [ST95]).

In our approximate search problem, we are given a set of words (vocabulary) and a query consisting of a query word  $T$  and tolerance value  $k$ , we are asked to find all words  $P$  in the vocabulary that matches the keyword  $T$  with at most  $k$  errors (that is, the edit distance between  $P$  and  $T$  should not be more than  $k$ ). We first find all words  $P$  in the vocabulary such that the  $q$ -gram distance between  $P$  and  $T$  is not more than  $2qk$  (i.e.,  $D_q(P, T) \leq 2qk$ ), then we apply the dynamic programming algorithm to  $P$  and  $T$ . By Lemma 4, we won't miss any qualifying words  $P$ . On the other hand, this filtering technique allows us to reduce the number of words in the vocabulary that need to be examined by the expensive dynamic programming algorithm.

## 4 Monotonous Bisector Trees

The data structure we choose to store the  $q$ -gram profiles of all words in the vocabulary is the so-called *monotonous bisector tree* (MBT) (cf. [NVZ92]).

**Definition 3 (monotonous bisector tree).** *Let  $S$  be a finite set of points in an arbitrary space  $E$  with distance function  $d$ , and let  $x_1 \in S$ . A monotonous bisector tree  $MBT(S, x_1)$  is a binary tree defined (recursively) as follows:*

*If  $1 \leq |S| \leq 2$ , then  $MBT(S, x_1)$  consists of a single node containing all elements of  $S$ .*

*If  $|S| > 2$ , then the root  $w$  of  $MBT(S, x_1)$  contains two points:  $x_1$  and another point  $x_2 \in S - \{x_1\}$ . The children of  $w$  are  $MBT(S_1, x_1)$  and  $MBT(S_2, x_2)$  such that  $S_i$  contains all points in  $S$  that are closer to  $x_i$  than to  $x_j$  ( $i, j = 1, 2$ ;  $i \neq j$ ) with respect to the distance function  $d$ . The points in  $S$  with equal distance to  $x_1$  and  $x_2$  are split into two halves, one for  $S_1$  and one for  $S_2$ .*

Note that by the definition  $S_1 \cap S_2 = \emptyset$  and  $S_1 \cup S_2 = S$ . The two subtrees  $MBT(S_1, x_1)$  and  $MBT(S_2, x_2)$  can be regarded as two clusters of  $S$ . For our

purpose, we additionally store in each node of the tree the radius of the cluster corresponding to that node:

$$\text{radius}(S_i, x_i) := \max\{d(x, x_i) | x \in S_i\}.$$

**Lemma 5 ([NVZ92]).**

1. In an MBT, the radii of the clusters corresponding to the nodes on a path from the root down to a leaf makes a monotonously decreasing sequence.
2. Let  $S \subset R^d$  be a finite set of  $n$  points. For any  $L_p$ -metric ( $1 \leq p \leq \infty$ ) an MBT with logarithmic height can be constructed in  $O(n \log n)$  time and uses  $O(n)$  storage, assuming that the distance between any two points can be computed in constant time.

In our case, the underlying space is the  $|\Sigma|^q$ -dimensional real space of the  $q$ -gram profiles under the  $L_1$ -metric. This high dimension does not adversely affect the efficiency of our methods, because the construction of the MBT is mainly based on distance relations between two points and can be carried out with the compact  $q$ -gram profiles.

### Fixed-Radius Near Neighbor Search

Monotonous bisector trees support a large variety of proximity queries. What we are interested in is the fixed-radius near neighbor search: we search for all points in  $S$  with distance at most  $r$  to a given point  $p \in E$ ;  $p$  is called the center and  $r$  the radius of the search ball. Starting from the root of the tree, suppose we are now at a node corresponding to the subtree  $MBT(V, x)$ . We can prune the subtree  $MBT(V, x)$  from our search if

$$D_q(p, x) - \text{radius}(V, x) > r.$$

We accept the points stored in the whole subtree if

$$D_q(p, x) + \text{radius}(V, x) < r.$$

Furthermore, we check if the search ball intersects the bisector of  $x_1$  and  $x_2$  or not. If the search ball lies in one side of the bisector we can restrict our search to the subtree corresponding to this side. More precisely, the search can be restricted to the subtree  $MBT(S_1, x_1)$  if

$$D_q(p, x_1) - D_q(p, x_2) < -2r, \text{ and correspondingly,}$$

the search can be restricted to the subtree  $MBT(S_2, x_2)$  if

$$D_q(p, x_2) - D_q(p, x_1) < -2r.$$

Extensive experiments have shown that the average query time for the fixed-radius near neighbor search is proportional to  $\log n + s$  where  $n$  is the number of points stored in the tree and  $s$  the size of the output of the query (cf. [NVZ92], [Zir90] and [Zir92]).

## 5 Constructing the Index

Let the vocabulary be  $V = \{P_1, P_2, \dots, P_z\}$  where each  $P_i$  is a string over some finite alphabet  $\Sigma$  (e.g., the English language alphabet). Let  $T$  be the query word, also a string over  $\Sigma$ . The basic idea is to preprocess the vocabulary into a monotonous bisector tree, to make subsequent searches with different query words fast.

### 5.1 Construction of the Index

The index is constructed in two steps.

1. Compute the set  $Q$  of the  $q$ -gram profiles of all words in the vocabulary  $V$ .
2. Construct the monotonous bisector tree  $MBT$  for the set  $Q$  under the  $q$ -gram distance.

The choice of a proper value for  $q$  – the number of characters in a  $q$ -gram – will depend on the particular application. For example, our calculation shows that in the Cystic Fibrosis Collection [SWWT91] the average size of a keyword is about 15 characters long (a phrase such as AMINO-ACID-METABOLISM-INBORN-ERRORS is treated as a single keyword in that collection). The Cystic Fibrosis Collection contains about 1500 documents indexed with the term 'cystic fibrosis' in the (USA) National Library of Medicine's MEDLINE database. So for a typical medical science document database,  $q$  may be chosen as  $q = 2$ , or, 3. In the case of DNA/RNA or protein databases, a keyword (pattern) can be hundreds or thousands of units (nucleotides or amino acids) long, so we should choose much larger  $q$  in such cases.

In a real approximate document retrieval application, we should associate each keyword  $P$  in the vocabulary with a pointer to the locations of the documents that contain this word. This augmented version of the vocabulary can be constructed using standard techniques such as *inverted file*. In other words, our index can be built on the top of the inverted file for the document database.

### 5.2 The Search Algorithm

Given a keyword (query word)  $T$  and a tolerance value  $k$ , we want to find all the documents in the database that contains the keyword  $T$  or any other word that matches  $T$  with at most  $k$  errors. The search is done in the following steps:

1. Compute the  $q$ -gram profile  $G(T)$  of  $T$ .
2. Select candidate words from the vocabulary  $V$  by performing a fixed-radius near neighbor search on the  $MBT$  of  $V$  with  $G(T)$  being the center and  $2qk$  the radius.
3. Check the candidate words obtained in the preceding step (Step 2) one by one against  $T$  using the dynamic programming algorithm.  
The result of this step is a set of matching words  $W$  that has an edit distance of at most  $k$  to  $T$ .

4. Locate all documents in the database that contain the matching words  $W$  obtained in the preceding step (Step 3) using some existing index built on the database (such as inverted file or bitmap index).

Note Step 2 is based on Lemma 4.

### 5.3 Updating the Index

Inserting a word  $P$  into the index is easy: we compute the  $q$ -gram profile  $G(P)$  of  $P$  and insert it into the  $MBT$  of the vocabulary. Deleting a word  $P$  from the vocabulary is done similarly.

### 5.4 The Complexity of the Index Structure

#### Cost of the Index

Let  $M$  denote the sum of the lengths of all words in the vocabulary  $V$  and  $z$  the number of words in the vocabulary  $V$ .

By Lemma 1, Step 1 in the construction of the index takes  $O(M)$  time. Since each time after we have computed a  $q$ -gram profile for a word in  $V$  we can insert it immediately into the  $MBT$ , so Step 1 only needs  $O(|\Sigma|^q)$  working space.

From Lemma 5 we know that Step 2 uses  $O(M \log z)$  time and  $O(M)$  storage.

Hence, the index needs  $O(M \log z)$  time to construct and it occupies  $O(M)$  storage.

#### Query Time

Let  $n$  be the length of the query word  $T$  and  $z$  the number of points in the  $MBT$  ( $z$  is equal to the number of words in the vocabulary  $V$ ).

Step 1 in the Search Algorithm takes  $O(n)$  time.

Since the expected query time for the fixed-radius near neighbor search on an  $MBT$  is  $t \log z + o$  where  $z$  is the number of points stored in the tree,  $o$  the size of the output of this fixed-radius near neighbor search and  $t$  the time needed to compute the distance between any two points (cf. [NVZ92], [Zir90] and [Zir92]), Step 2 needs time  $O(n(\log z + b))$  on the average where  $b$  is the number of points in the tree contained by the  $L_1$ -ball with the  $q$ -gram profile  $G(T)$  being the center and  $2qk$  the radius.

If the dynamic programming algorithm is used, Step 3 takes  $O(bn^2)$  time.

Thus, the search algorithm will need  $O(n(\log z + bn))$  time on the average (This time does not include the time needed by Step 4 which uses some existing index built on the database for exact search operations).

**Experiment.** We wrote a program to do the following experiment on the Cystic Fibrosis Database [SWWT91]: we randomly choose a word  $W$  from the vocabulary  $V$  of about 2100 keywords. Then we randomly choose  $k$  positions in  $W$  and change the characters in these positions to other random characters. The resulting word is then become our query word  $Q$ . Then we search the vocabulary  $V$  for keywords whose edit distance to  $Q$  is not more than  $k$  – the number of

such matching keywords is then  $c$ , and we search for keywords whose  $q$ -gram distance to  $Q$  is not more than  $2kq$  – the number of such matching keywords is then  $b$ . We repeat the above experiment thousands of times and found out that on the average the ratio of  $b$  to  $c$  is about 1.2 - 11 for  $k = 2$  and  $q = 2$ . For bigger  $k$ , the ratio is higher. That means that for each keyword in the vocabulary that matches our query word with at most  $k$  errors (i.e., an edit distance of at most  $k$ ), we need to examine about 1.2 - 11 keywords in the vocabulary on the average for small  $k$ .

### Updating Time

Inserting/deleting a word of length  $m$  into/from the *MBT* of the vocabulary takes  $O(m \log z)$  time.

## 6 Conclusion and Future Work

We have presented an index structure for approximate document retrieval. This index allows us, for each query issued by a user that consists of a keyword  $T$  and a tolerance value  $k$ , to quickly find the locations of all the documents in the database that contain the keyword  $T$  or any other word that matches  $T$  approximately (We say that two words  $X$  and  $Y$  match each other approximately if their edit distance does not exceed  $k$ , the tolerance value specified by the user; typically a user will choose  $k = 1, 2, 3$ , or  $4$ ). Computation of the edit distance between two strings  $X$  of length  $n$  and  $Y$  of length  $m$  needs  $\Omega(nm)$  time. Thus, it could take us enormous amount of time if we had to examine every word in the vocabulary against the keyword  $T$ . Our index structure allows us to examine only a small subset of the words in the vocabulary that are likely to match (approximately) the keyword, thus reducing the search time significantly. This filtering technique is based on the  $q$ -gram function, which can be computed in linear time. Preliminary experiments have shown that this filtering technique is very effective. The index is stored in a data structure called the monotonous bisector tree (*MBT*).

More sophisticated *approximate* boolean queries can also be implemented using this index.

In applications that involve enormous large vocabularies, one should consider using a *B*-tree version of the *MBT* tree to hold the index.

We are currently implementing our index structures and will empirically compare our index structures with other existing index structures using some real world data such as the TREC collections.

We are also considering extending our technique so that we can quickly answer approximate search queries in DNA/RNA or protein sequence databases (such as Genbank and SwissProtein).

## References

- [AHU76] A. V. Aho, D. S. Hirschberg and J. D. Ullman, Bounds on the Complexity of the Longest Common Subsequence Problem, *Journal of the ACM*, vol.23, No.1, January 1976, pp. 1-12.

- [NVZ92] H. Noltmeier, K. Verbarg and C. Zirkelbach, Monotonous bisector\* trees - a tool for efficient partitioning of complex scenes of geometric objects. In *Data Structures and Efficient Algorithms: Final Report on the DFG Special Joint Initiative*, Vol. 594 of L.N.C.S., Springer-Verlag, 1992.
- [OM88] O. Owolabi and D. R. McGregor: Fast approximate string matching. *Software - Practice and Experience* 18(4) (1988), 387-393.
- [Sha48] C. E. Shannon, A mathematical theory of communications. *The Bell Systems Techn. Journal* 27 (1948), 379-423.
- [SK83] (eds.) D. Sankoff and J. B. Kruskal, *Time Warps, String Edits, And Macromolecules: The Theory And Practice Of Sequence Comparison*, Addison-Wesley Publishing Company, Inc., 1983.
- [ST95] E. Sutinen and J. Tarhio, On Using  $q$ -gram Locations in Approximate String Matching, in *Proc. European Symposium on Algorithms*, 1995, LNCS 979, Springer-Verlag, pp. 327-340.
- [SWWT91] W. M. Shaw, J. B. Wood, R. E. Wood, and H. R. Tibbo, The Cystic Fibrosis Database: Content and Research Opportunities, in *Library and Information Science Research*, 13:347-366, 1991.
- [Ukk92] E. Ukkonen, Approximate string matching with  $q$ -grams and maximal matches. *Theoretical computer Science* 92 (1992), pp. 191-211.
- [WC76] C. K. Wong and A. K. Chandra, Bounds for the string editing problem. *Journal of the ACM*, vol.23, No.1, January 1976, pp. 13-16.
- [Zir90] C. Zirkelbach, Monotonous bisector trees and clustering problems, Report, Department of Computer Science, University of Würzburg, Germany, 1990.
- [Zir92] C. Zirkelbach, Geometrisches Clustern - ein metrischer Ansatz, Dissertation, Department of Computer Science, University of Würzburg, Germany, 1992.



# Apply Feedback Control Theory to Design Soft Real-Time Search Engine System

Huayong Wang and Yiqi Dai

Department of Computer Science and Technology  
Tsinghua University, Beijing 100084, P.R.China  
wanghy02@mails.tsinghua.edu.cn, dyq@theory.tsinghua.edu.cn

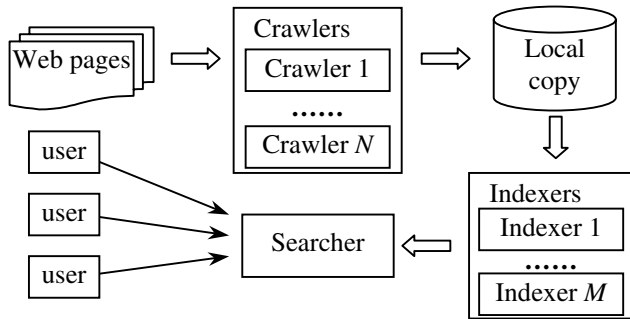
**Abstract.** This paper proposes a design method for soft real-time search engine system, and provides proofs to its correctness and robustness both in control theory and by practical experiments. An analyzable mathematical model is set up to approximately describe the nonlinear and time-varying search engine system. The feedback control theory is adopted to prove the system's stableness, zero steady state error and zero overshoot. The soft real-time guarantee is satisfied while the feedback system is in stable state. The experiment results further prove the effectiveness of our scheme.

## 1 Introduction

Quite a lot of literatures on search engine system have been presented in the past few years. However, substantial opportunities for improving the performance of web search still exist, and research in this field continues to expand rather than shrink. In this paper, the runtime performance of search engine system in the Internet environment is concerned, and a general framework to design search system with soft real-time guarantee is proposed.

Most practical and commercially operated search engines are based on a centralized architecture that relies on a set of key components, namely Crawler (or Robot), Indexer, and Searcher. A crawler is a module aggregating documents from WWW in order to generate a local copy. An indexer is a module that takes a collection of documents or data and builds a searchable index from them. Common methods are inverted files, vector space, suffix structures and hybrids of these. A searcher is working on the output files from the indexer. The searcher accepts user queries, executes the query over the index, and returns sorted search results back to users. Figure 1 illustrates the architecture of the search engine system.

In order to keep the local copy up to date, it is desirable for search engine system to have soft real-time property. Real-time systems are computing systems that must react within precise time constraints to events in the environment [1]. As a consequence, the correct behavior of these systems depends not only on the value of the computation but also on the time at which the results are produced. The time before which a task should be completed to avoid damage to the system is called deadline. A system is said to be soft real-time if deadline missing decreases the performance of the system, but does not jeopardize the system's correct behavior.



**Fig. 1.** The architecture of search engine system

Although there exists much research work that evaluates and improves the performance of search engine system, very limited work has been done to propose design methods for real-time search system, and even less work provides theory analysis. The difficulty lies on the fact that the search engine system is a nonlinear and time-varying system. Therefore it is not easy to describe it by a mathematical model. Without an analyzable mathematical model, it is impossible to apply real-time theory to the system design with proofs. Most current methods to improve the efficiency of search system are to run more crawlers, both in the centralized and distributed architectures. These methods cannot provide precisely control to the network bandwidth required for the search operation. Therefore bandwidth waste is usually inevitable. How to provide real-time guarantee without relying on over-provisioning of network bandwidth becomes a great challenge. The contribution of this paper is to resolve these problems. After this introduction, section two introduces the related works in the field of search engines and real-time scheduling algorithms. Section three explains our system design methods in theory. Section four gives the performance evaluation. Section five makes the conclusion.

## 2 Related Work

It is an important requirement for search engines to provide most fresh information to users. Much of the recent work has been done to reduce the update cycle and improve the system efficiency. For example, agent technology has been tried to implement distributed search in paper [2]. This kind of system runs multiple agents locally in different websites, and sends a compact summary back to the search engine. An AS-based distributed system is proposed in paper [3], which divides the search domain into small blocks to improve the system throughput. Nobuyoshi Sato develops a distributed search engine, called Cooperative Search Engine (CSE), in order to retrieve fresh information [4, 5]. In CSE, a local search engine located in each web server makes an index of local pages. And a meta search server integrates these local search engines in order to realize a global search engine. Paper [6] describes a search system in P2P network. Paper [7] advocates an approach in which web servers themselves track the changes happening to their content files for propagating updates to search engines. However, all these systems do not take the real-time property into consideration. The runtime behavior is uncertain and cannot be guaranteed in theory.

Paper [8] proposes a scheme for real-time search engine. In this system, users can specify an explicit deadline, and the search results will be returned before the deadline. This system provides real-time guarantee for user queries, not for the searched websites. It just tries best to search web pages as many as possible in the specified time interval. So it is not suitable for large-scale search. J.Talim discusses how to control the number of crawlers using the tools of stochastic process [9]. This research inspires us very much while we build the mathematical model of search engine system. It is generally believed that changing resource allocated to search engines, such as the number of crawlers, will change the search ability of the system.

On the other hand, it is necessary to introduce some related work in the field of real-time system. The field of real-time scheduling has been growing exponentially since the mid-1970s. Liu and Layland studied the problem of scheduling hard tasks in 1973 [10]. From then on, many of the classical algorithms and analysis methods focus on strict hard deadlines, by which a system is deemed schedulable only if every task is guaranteed to complete before its deadline. For example, the famous RM, DM, EDF algorithms have been proved to be optimal under their respective conditions [1]. In these researches, a real-time system is modeled as a set of  $N$  independent tasks  $S = \{\tau_1, \dots, \tau_N\}$ . Each task consists of an infinite sequence of identical activities that are regularly or irregularly activated, called instances or jobs. The  $j$ th job of the  $i$ th task will be denoted by  $J_{ij}$ . If the jobs are activated regularly, the task is called periodic task; otherwise, it is called aperiodic task.

Further research shows that the traditional scheduling algorithms are highly inefficient in the unpredictable environment, where the execution time of a task varies greatly from one instance to another. The experiment of T.S.Tia demonstrates that traditional scheduling methods lead to computation resource loss, and the processors are very much under-utilized, even to 13% - 25% [11]. In order to overcome the shortcoming of low efficiency, flexible scheduling algorithms are proposed. One of them is stochastic scheduling [12]. Stochastic scheduling does not guarantee each job to complete before its deadline, it only promises that a certain percent of jobs will complete in time. So it is a kind of soft real-time scheduling algorithm.

In recent years, much research work has applied control theory to computing systems in order to achieve acceptable performance in unpredictable environment. For example, several papers [13, 14, 15] present flexible scheduling techniques to improve digital control system performance. A control-theoretical approach has also been applied to provide Qos guarantees in web servers [16, 17] and e-mail servers [18]. Feedback control outperforms the open loop control in unpredictable environment. Feedback control theory is adopted to implement adaptive real-time system in paper [19, 20, 21]. However, feedback control theory has been mostly applied in mechanical and electrical systems. In trying to apply feedback control theory to a search engine system, the modeling and implementation face significant research challenges. Some of these challenges will be answered in this paper.

### 3 System Design with the Guide of Control Theory

Let us explain how to define a search engine as real-time system. For a search engine system, let us assume there are  $L$  websites in its search domain, which correspond

to  $L$  periodic tasks. Each visitation to a website is deemed as a job. A search engine employs different update cycles for different websites in the search domain. That is to say, to keep up with the update speed of a website, the search system is required to visit this website in a fixed time interval. Therefore, the time interval can be deemed as period  $T$ , the end of the interval can be deemed as deadline. The purpose of our design method is to implement a soft real-time search engine, which can guarantee the temporal constraint for a certain percent of websites. And this percent value is adjustable by system implementers.

### 3.1 Variables in the Control System

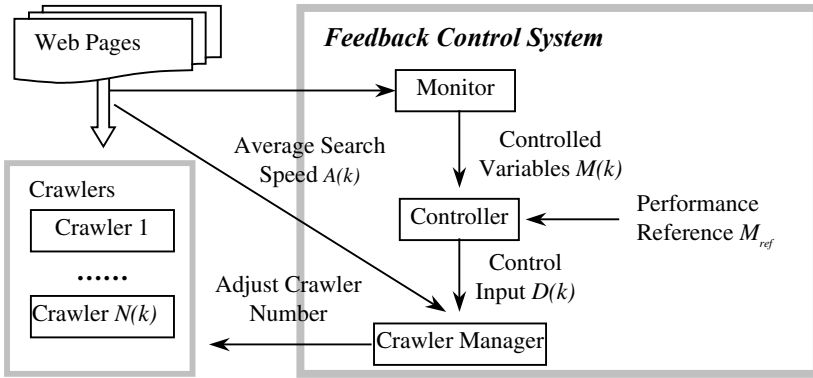
The first step in designing the real-time search system is to decide the following variables in terms of control theory. Readers can refer to [22, 23, 24] for the basic knowledge of control theory.

**Controlled variable** is the performance metric controlled by the real-time search engine system. In our model, the variable deadline satisfied ratio  $M(k)$  is used as controlled variable, which is defined over a time window  $\{(k-1)W, kW\}$ , where  $W$  is the sampling period and  $k$  is called the sampling instant. For the simplicity, time window  $\{(k-1)W, kW\}$  is also called time window  $k$ . The deadline satisfied ratio  $M(k)$  at the  $k$ th sampling instant is defined as the ratio between the number of the jobs that respect their deadlines and the total number of all jobs (completed and aborted) in the sampling window  $k$ .

**Performance reference** represents the desired system performance in term of the controlled variable. For example, a particular system may require a deadline satisfied ratio  $M_{ref} = 90\%$ . The difference between a performance reference and the current value of the controlled variable is called an error. The deadline satisfied ratio error  $E_M(k) = M_{ref} - M(k)$ .

**Manipulated variable** is system attribute that can be dynamically changed by the system to affect the value of the controlled variable. In our system, the manipulated variable is search speed  $S(k)$ . The search speed  $S(k)$  at the  $k$ th sampling instant is defined as the product of  $N(k)$  and  $A(k)$ , where  $N(k)$  is the number of crawlers at time window  $k$ ,  $A(k)$  is the average search speed of one crawler at time window  $k$ .  $A(k)$  can be measured by recording the average number of web pages visited by one crawler in time window  $k$ , or the number of http sessions in time window  $k$ .  $A(k)$  is time-varying and dependent on the quality of the network. However, it is assumed that  $A(k)$  does not vary greatly in two adjacent time windows.

Figure 2 illustrates the framework of our real-time search engine system. The monitor measures the controlled variable  $M(k)$  and feeds the samples back to the



**Fig. 2.** Framework of real-time search engine system

controller. The controller compares the performance references  $M_{ref}$  with  $M(k)$  to get the current error  $E_M(k)$ , and computes a change  $D(k)$  (called control input) to the search speed  $S(k)$ . The controller uses a simple P (proportional) control function to compute the correct search speed value to compensate for the disturbance and keep the deadline satisfied ratio close to the reference  $M_{ref}$ . The crawler manager dynamically changes the search speed at each sampling instant  $k$  according to the control input  $D(k)$  by adjusting the crawler number. The goal of the crawler manager is to enforce the new search speed  $S(k+1) = S(k) + D(k)$ . In our system, crawler manager can calculate the new crawler number by formula

$$N(k+1) = \lceil S(k+1) / A(k) \rceil \quad (1)$$

Formula (1) is feasible because of the previous assumption that the average search speed of one crawler does not vary greatly in two adjacent time windows.

### 3.2 Open Loop Transfer Function

The second step is to establish an analytical model to approximate the search system. Although it is difficult to precisely model a nonlinear and time-varying system, we can approximate such a system with a linear model for the purpose of control design because of the robustness of feedback control with regard to system variations. Starting from the control input, the crawler manager changes the search speed at every sampling instant  $k$ .

$$S(k+1) = S(k) + D(k) \quad (2)$$

Since formula (1) uses  $A(k)$  to compute  $N(k+1)$ , the precise value of actual search speed at time window  $k+1$   $AS(k+1)$  may differ from the predicted value  $S(k+1)$ .

$$AS(k+1) = R_{AS}(k+1)S(k+1)$$

We can also rewrite it as following:

$$AS(k) = R_{AS}(k)S(k) \quad (3)$$

where  $R_{AS}(k)$  is a time-variant variable that represents the extent of search speed variation in terms of the predicted value. Let  $R_{AS} = \max\{R_{AS}(k)\}$ , which is called the worst-case extent of search speed variation. Hence Equation (3) can be simplified to the following formula for the purpose of control design:

$$AS(k) = R_{AS}S(k) \quad (4)$$

$M(k)$  usually increases nonlinearly with the search speed  $AS(k)$ . The relationship between  $M(k)$  and  $AS(k)$  needs to be linearized by taking the derivative at the vicinity of the performance reference  $M_{ref}$  as the deadline satisfied ratio factor  $R_M$ :

$$R_M = \frac{dM(k)}{dAS(k)} \quad (5)$$

In practice, the deadline satisfied ratio factor  $R_M$  can be estimated experimentally by plotting a  $M(k)$  curve as a function of  $AS(k)$  based on experimental data. Now, we have the following linearized formula:

$$M(k) = M(k-1) + R_M(AS(k) - AS(k-1)) \quad (6)$$

Based on Equations (2) and (4), the analytical model for the deadline satisfied ratio output is as following:

$$M(k) = M(k-1) + R_M R_{AS} D(k-1) \quad (7)$$

We now convert the model to z-domain transfer function that is suitable to control theory methods. Let  $Y(z)$  be the z-transform of the output variable  $M(k)$ , and  $X(z)$  be the z-transform of the input variable  $D(k)$ . A linear system can be represented by a transfer function  $H(z) = Y(z)/X(z)$ . For our system, formula (7) can be converted into z-domain as  $Y(z) = z^{-1}Y(z) + R_M R_{AS} X(z)z^{-1}$ . So the transfer function in open loop is:

$$H_{open}(z) = \frac{R_M R_{AS}}{z-1} \quad (8)$$

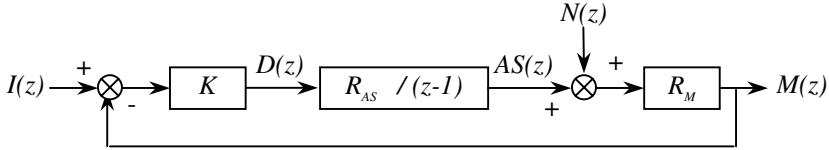
### 3.3 Closed-Loop Transfer Function

The third step is to establish the closed-loop transfer function. At each sampling instant  $k$ , the controller computes a control input  $D(k)$  based on the deadline satis-

fied ratio error  $E_M(k)$ . We choose a simple P (proportional) control function to compute the control input.

$$D(k) = K \times E_M(k) \quad (9)$$

It is obvious that the parameter  $K$  has great impact on the system performance. In order to analyze the suitable value for parameter  $K$ , we convert figure 2 to the following block diagram according to control theory. In figure 3, we introduce the closed loop, and the closed-loop system transfer function should be recalculated based on the open loop one.



**Fig. 3.** Feedback control loop

The system input  $I(z)$  is the performance reference, which is modeled as a step signal,  $M_{ref}z/(z-1)$  in the  $z$ -domain. From control theory, we can establish a closed-loop transfer function of deadline satisfied ratio in response to the reference input:

$$H_1 = \frac{KR_{AS}R_M}{(z-1 + KR_{AS}R_M)} \quad (10)$$

and the output can be computed by:

$$Y(z) = H_1(z)I(z) = \frac{KR_{AS}R_M}{z - (1 - KR_{AS}R_M)} \times \frac{M_{ref}z}{z-1} \quad (11)$$

The second input to the closed-loop system is the disturbance  $N(z)$  that adds to the total requested search speed  $AS(z)$ . For example, some websites suddenly have huge body of new information that cost the crawlers much more time to process than expected. This kind of sudden change represents more severe variation than the other smooth ones, so we can model  $N(z)$  as a step signal that jumps instantaneously from 0 to  $N$ , or  $N(z) = Nz/(z-1)$  in the  $z$ -domain. Here the value  $N$  means the extent of the disturbance. Considering the disturbance input, the closed-loop system transfer function for the deadline satisfied ratio in response to the disturbance is as following:

$$H_2 = \frac{R_M(z-1)}{(z-1 + KR_{AS}R_M)} \quad (12)$$

The system output while considering both reference input and disturbance input can be computed by:

$$Y(z) = H_1(z)I(z) + H_2(z)N(z) \quad (13)$$

### 3.4 System Profile

According to control theory, a system is stable if and only if all the poles of its transfer function are in the unit circle of  $z$ -plane. The only one pole in formula (10) and (12) is  $1 - KR_{AS}R_M$ . So, if  $0 < K < 2/R_{AS}R_M$ , the stability can be guaranteed.

In our system,  $M(k)$  is the output, and its corresponding format in  $z$ -domain is  $Y(z)$ . The final value theorem of digital control theory states that the system output converges to a final value:  $\lim_{k \rightarrow \infty} M(k) = M(\infty) = \lim_{z \rightarrow 1} (z-1)Y(z)$ . From the formula (13),

$$\begin{aligned} & \lim_{z \rightarrow 1} (z-1)Y(z) \\ &= \lim_{z \rightarrow 1} [(z-1) \left( \frac{KR_{AS}R_M}{z-1+KR_{AS}R_M} \times \frac{M_{ref}z}{z-1} + \frac{R_M(z-1)}{(z-1+KR_{AS}R_M)} \times \frac{Nz}{z-1} \right)] \\ &= M_{ref} \end{aligned}$$

This means that the steady state error is zero.

According to control theory, for the system transfer function (formula 10), the overshoot remains zero in response to reference input if the closed loop pole equals to or larger than zero. So, if the parameter  $K$  satisfies the condition  $0 < K \leq 1/R_{AS}R_M$ , the deadline satisfied ratio  $M(k)$  reaches zero overshoot. The settling time of this feedback system can be adjusted by parameter  $K$ , because the parameter  $K$  has direct relation to the pole of the transfer function. It is a basic technique to determine the settling time according to poles in a first order system in control theory.

While the search system is steady, the output  $M(k)$  will be stabilized at the value  $M_{ref}$ . When disturbance occurs, the system will converge to the desired value  $M_{ref}$ . The speed of this converging can be adjusted by parameter  $K$ . If  $M(k)$  is stabilized at the value  $M_{ref}$ , the soft real-time guarantee is satisfied according to the concept of stochastic scheduling.

## 4 Performance Evaluations

The hardware platform of our experiments is a cluster of four PCs with CPU800MHz and memory 256M. Multiple computers are involved in order to eliminate the hardware bottleneck to the number of processes. Usually, when the number of processes is very large, the efficiency of multiple processes cannot be achieved in a single CPU machine, because those processes are running concurrently, not in parallel. The search domain is limited in 1200 Chinese news web pages (like <http://news.sina.com.cn>), which provide continually changed fresh news everyday. Some websites changed



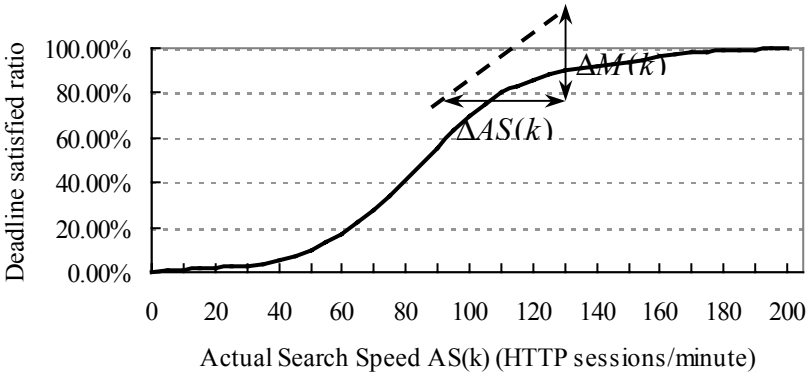
frequently, and the shortest update period is 10 minutes; some changed less frequently, and the longest update period is 3 hours. The search engine is designed to receive all the news in these websites in the respective update periods and with lowest possible network bandwidth. The total news is about 30000 pieces each day, except for Sunday and Saturday. The following table records the update period distribution of the test set.

**Table 1.** Distribution of the update periods

Number of websites	Update Period
18	10 minutes
273	30 minutes
562	1 hour
258	2 hour
89	3 hours

The first step is to determine the relation between deadline satisfied ratio  $M(k)$  and actual search speed  $AS(k)$ . The relation is nonlinear partially because the uneven distribution of deadlines in our test set. It is more difficult to provide real-time guarantees to tasks with short deadlines. Figure 4 shows the experiment results of  $M(k)$  based on  $AS(k)$ . Here,  $AS(k)$  is measured by http sessions. The HTTP header fields, such as Last\_Modified and Content-Length, are able to facilitate search operations. So, a certain number of operations of crawlers do not involve the visitations to html pages. If  $M_{ref} = 80\%$ , we can draw the tangent line at point

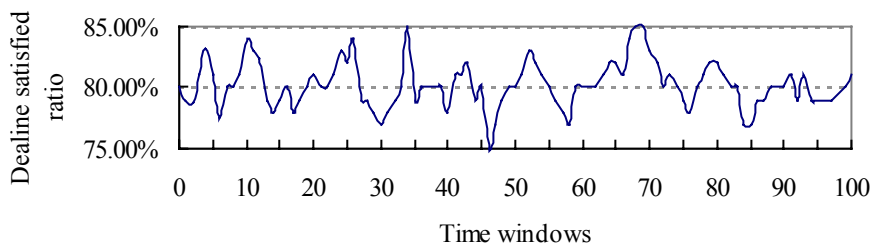
$M(k) = 0.8$ . The tangent slope is  $R_M = \frac{dM(k)}{dAS(k)}$ . Therefore  $R_M(0.8) = 0.01$ .



**Fig. 4.**  $M(k)$  curve based on  $AS(k)$

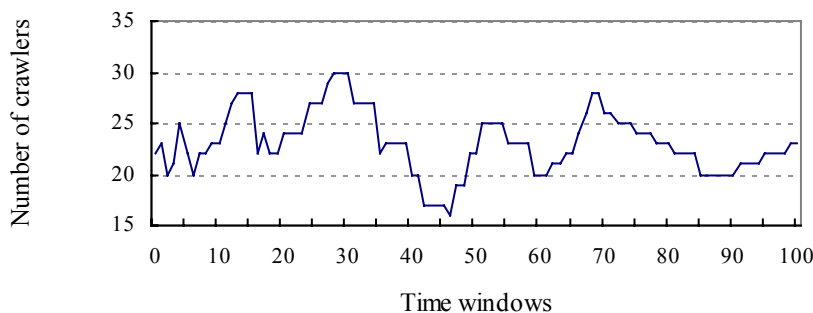
In the later steps, we use  $R_{AS} = 1.5$ ,  $M_{ref} = 80\%$  and sampling period  $W = 3$  minutes. Note that the point  $M(k) = 80\%$  is more difficult to control because it has relatively higher curvature in figure 4, which makes it least amenable for linear

modeling. Figure 5 records the system performance in a segment of 300 minutes during stable state. While the disturbance exists, the variation range of deadline satisfied ratio is within 5% to  $M_{ref}$ .



**Fig. 5.** Stable state performance

Figure 6 records the change of crawler number during the same time segment. In this segment, the maximum number is 30, and minimum number is 16. With the real-time guarantee, the crawlers are dynamically increased or decreased to save the network bandwidth, which is one of the advantages of our system. Traditional search engine system cannot provide this kind of control between  $M(k)$  and network bandwidth usage. System implementers can choose a reasonable tradeoff point between these two variables.



**Fig. 6.** Chang of crawler number

## 5 Conclusions

This paper proposes a design method for soft real-time search engine system. A mathematical model of search system is set up. And the control theory is applied to prove the system performance guarantee. Experiment results further prove the effectiveness of our scheme. Future work will further investigate the feedback control of crawlers and integrate the indexer and searcher into the real-time framework.

## References

1. G. Buttazzo. *Hard Real-Time Computing System: Predictable Scheduling Algorithms and Applications*. Kluwer Academic Publishers, Massachusetts, 2000.
2. Prasanna Thati, Po-Hao Chang, and Cul Agha. *Crawlets: Agents for High Performance Web Search Engines*. Lecture Notes of Computer Science 2240, Springer-Verlag, pp. 119-134, 2001.
3. Zhang Xiaohui, Wang Huayong, Chang Guiran and Zhao Hong. An autonomous system-based distribution system for web search. In proceedings of IEEE International Conference on Systems, Man, and Cybernetics, Vol. 1, Pages 435-440, 2001.
4. Nobuyoshi Sato, Minoru Uehara, Yoshifumi Sakai, and Hideki Mori. Distributed Information Retrieval by using Cooperative Meta Search Engines. In proceedings of the 21st IEEE International Conference on Distributed Computing Systems Workshops (Multimedia Network Systems, MNS2001), pp. 345-350, 2001.
5. Nobuyoshi Sato, Minoru Uehara, Yoshifumi Sakai, Hideki Mori. Fresh Information Retrieval using Cooperative Meta Search Engines. In proceedings of the 16th International Conference on Information Networking (ICOIN-16), Vol.2, pp. 7A-2-1-7, 2002.
6. Herwig Unger and Markus Wulff. Towards a Decentralized Search Engine for P2P-Network Communities. In proceedings of 11th Euromicro Conference on Parallel, Distributed and Network-Based Processing, pp. 429-499, 2003.
7. Vijay Gupta and Roy Campbell. Internet Search Engine Freshness by Web Server Help. In proceedings of 2001 Symposium on Applications and the Internet, pp. 113-119, 2001.
8. Augustine Chidi Ikeji, and Farshad Fotouhi. An Adaptive Real-Time Web Search Engine. In proceedings of the second international workshop on Web information and data management, 1999.
9. J.Talim, Z.Liu, Ph.Nain, and E.G.Coffman. Controlling the Robots of Web Search Engines. ACM SIGMETRICS Performance Evaluation Review, Vol. 29, Issue 1, June 2001.
10. C.L. Liu, and J.W. Layland. Scheduling Algorithms for Multiprogramming in a Hard Real Time Environment. *Journal of the ACM*. Vol. 20, No. 1, pp. 46-61, 1973.
11. T.-S. Tia, Z. Deng, M. Shankar, M. Storch, J. Sun, L.-C. Wu, and J.W.-S. Liu. Probabilistic Performance Guarantee for Real-Time Tasks with Varying Computation Times. *IEEE Real-Time Technology and Applications Symposium*. pp. 164-173, 1995.
12. J.L. Diaz, D.F. Garcia, K. Kim, C.G. Lee, L. Lo Bello, J.M. Lopez, S.L. Min, and O. Mirabella. Stochastic Analysis of Periodic Real-Time Systems. *IEEE 23rd Real-Time Systems Symposium*. pp. 289-300, 2002.
13. T.F. Abdelzaher, E.M. Atkins, and K.G. Shin. Qos negotiation in real-time system and its application to automatic flight control. In *IEEE Real-Time Technology and Application Symposium*, 1997.
14. G. Buttazzo, G. Lipari, and L. Abeni. Elastic task model for adaptive rate control. In *IEEE Real-Time System Symposium*, pp. 286-295, 1998.
15. M. Caccamo, G. Buttazzo, and L. Sha. Capacity sharing for overrun control. In *IEEE Real-Time Systems Symposium*, 2000.
16. T.F. Abdelzaher, and N. Bhatti. Web server Qos management by adaptive content delivery. In *International Workshop on Quality of Service*, 1999.
17. C. Lu, T.F. Abdelzaher, J.A. Stankovic, and S.H. Son. A feedback control approach for guaranteeing relative delays in web servers. In *IEEE Real-Time Technology and Application Symposium*, 2001.
18. S. Parekh, N. Gandhi, J.L. Hellerstein, D. Tilbury, T.S. Jayram, and J. Bigus. Using control theory to achieve service level objectives in performance management. In *IFIP/IEEE International Symposium on Integrated Network Management*, 2001.
19. L. Abeni, L. Palopoli, G. Lipari and J. Walpole. Analysis of a Reservation-Based Feedback Scheduler. In *Proceedings of IEEE 23rd Real-Time Systems Symposium*, pp. 71-80, 2002.

20. Lui Sha, Xue Liu, Ying Lu and Tarek Abdelzaher. Queueing Model Based Network Server Performance Control. In Proceedings of IEEE 23<sup>rd</sup> Real-Time Systems Symposium, pp. 71-80, 2002.
21. Anton Cervin, Johan Eker, Bo Bernhardsson, and Karl-erik Arzen. Feedback-Feedforward Scheduling of Control Tasks. Real-Time Systems, Vol. 23, Issue 1/2, pp. 25-53.
22. Cheng Shengtian, Guo Baolong, Li Xuewu and Feng Zhongzhe. Signal and System. Press of Xian Electronic Technology Univ., Xian, P.R.China, 2001.
23. Shun Zhengqi. System analysis and control. Tsinghua univ. Press, Beijing, P.R.China, 1994.
24. G.F. Franklin, J.D. Powell and M.L. Workman. Digital Control of Dynamic Systems (3<sup>rd</sup> Edition). Addison-Wesley, 1998.

# User Interest Detection on Web Pages for Building Personalized Information Agent

Yin Liu<sup>1,2</sup>, Wenyin Liu<sup>1</sup>, and Changjun Jiang<sup>2</sup>

<sup>1</sup> Department of Computer Science, City University of Hong Kong,  
Tat Chee Avenue, Hong Kong, PRC  
{liuyin, csluwy}@cityu.edu.hk

<sup>2</sup> Department of Computer Science, Tongji University,  
SiPing Road, Shanghai, PRC  
cjjiang@online.sh.cn

**Abstract.** In this paper we propose a vision-based approach to detecting user's interests on web pages. This approach first splits a web page into several blocks of topics according to visual cues and then analyzes the layout structure of the page based on these blocks. The user's interests are detected according to his browsing behaviors on the blocks and the layout structure of the web page. A personalized information agent is also developed to automatically extract new content from those web pages according to the user's interests. Our experiments show that our proposed approach to detecting user's interests on web pages achieves satisfactory accuracy and the personalized information agent significantly saves user's browsing time and improves user's browsing experience.

## 1 Introduction

Currently, the Web has become an important source of new information in our daily life. In order to attract users with different interests, the editors of those web sites usually put a lot of content, which belongs to different topics, on the same web page. However, when a particular user browses this page, he is usually interested in a small part of them only. If we can detect those topics in which the user is interested and retrieve updated content of these topics automatically from the web site, it will significantly save user's browsing time and improve user's browsing experience. This is the initial motivation of this paper.

In this paper, we propose a novel approach to this problem, which takes advantage of knowledge from two aspects. The first is the knowledge from the editor of the web page. When an editor creates a web page, he usually groups content of the same topic in one block with visually consistent appearance. Therefore, difference in appearance often indicates difference in topics. The second is the knowledge from the user who is browsing the page. If a user is interested in a topic, his click actions should often fall into the block which represents that topic. With such knowledge, it is possible for us to detect blocks in which the user is interested on a web page.

Our approach to detecting user's interests and extracting user-interested content on web pages is composed of three phases, i.e., Page Segmentation, Layout Analysis and Interest Detection. First, we split a web page into several blocks according to the difference in appearance. After splitting, each block represents a topic on the web page and the content in the same block should belong to the same topic. Second, we analyze layouts of the web page in consecutive days and match blocks between them in order to track changes of content of each topic. Third, we record user's browsing behaviors and detect the blocks in which the user is interested. Based on the detection result, new content in these user-interested blocks will be extracted and presented to the user.

In this paper, our major contributions are:

1. A vision-based approach is introduced into the personalized information filtering area, which can detect user's interests in a smaller granularity than existing approaches.
2. A novel approach is proposed to match layouts and track content change of the same web page over the time.

The rest of this paper is organized as follows. Section 2 presents a brief introduction to related works. Section 3 describes the details of the page segmentation algorithm. Section 4 discusses layout analysis algorithm for topics tracking. Section 5 describes the system architecture of our personalized information agent and the corresponding method to detect user's interest. The experiment result is presented and analyzed in Section 6. Finally we present a brief concluding remark in Section 7.

## 2 Related Works

Personalized information filtering has been in research for many years. The aim is to retrieve useful information according to user's interest. A lot of approaches have been proposed and many of them are NLP (Nature Language Processing) based, such as [1],[5]. They usually summarize key words from pages that the user has browsed to detect user interest. New pages are retrieved according to their text similarity to those key words. However, due to current limitations of NLP technologies, the accuracy of such approach is usually not satisfactory. We think not only the information from the text, but also the information from the visual appearance should be used to detect user interest. This is the major difference between our approach and existing NLP-based approaches.

In order to detect user's interest in blocks instead of the whole page, it is necessary for us to analyze the structure of web pages. Some researchers propose to detect content structure according to the structure of the DOM [6] tree of web pages, such as [2],[8]. Moreover, inspired by traditional image-based document analysis, in [3], a vision-based approach has been proposed to extract content structure from web pages. It is a top-down approach and the granularity of blocks in the final segmentation result is relatively large because it completely depends on the explicit and implicit separators on the web pages. The approach in [4], which is a little bit similar to ours,

first decomposes a web page into several basic blocks and then groups them together. However, the structure information in the DOM tree is not used and hence, the segmentation result mainly depends on the definition of similarity of blocks, which is sometime misleading.

### 3 Page Segmentation

The goal of the page segmentation algorithm is to split a web page into several blocks, each of which belongs to only one topic. As we mentioned above, when a web page is composed, the editor usually puts content, which belongs to the same topic, into a visually distinguishable block. This is the basic assumption of our algorithm. In order to make a block visually distinguishable on the page, the editor usually inserts empty blocks between this block and its neighbors to separate them apart, or make adjacent blocks look in different appearance. Based on these observations, we first decompose a web page into several salient blocks (refer to Section 3.1) and then cluster them into several topic blocks (refer to Section 3.2) according to their similarity in appearance. The final output of the page segmentation algorithm is the layout structure of the web page.

#### 3.1 Salient Block Decomposition

Our proposed page segmentation algorithm is based on the DOM [6] representation of web page. However, a lot of nodes in the DOM tree of the web page do not occupy any space on the page. Moreover, because the HTML language is mainly a rendering language, same appearance can be rendered by DOM trees of different hierarchical structures. Therefore, our algorithm first takes three steps to decompose a DOM tree into several basic units, which are also referred to as “salient blocks”, to obtain a compact visual representation of a web page.

The first step of salient block decomposition is to remove null nodes, which do not contain any spatial information, from the DOM tree to obtain a compact tree presentation. Two kinds of nodes are regarded as null nodes. The first is the node that does not occupy any space on the web page. The second is the node that has some children and the space it occupies is not larger than its children’s space. For the latter, it will be removed from the DOM tree and its children become children of its parent.

After removal of null nodes, we have obtained a compact DOM tree and each node in it contains some spatial information. However, it is not necessary for us to decompose the page to the level of the leaves in the DOM tree because many high level nodes are already visually consistent and distinguishable and should not be split further. In addition, not all nodes in the compact DOM tree contain useful information. Some blocks are inserted by the editor to separate other meaningful blocks. These blocks are referred to as separators. The second step of salient block decomposition is to detect those separators in the compact DOM tree using the following rules.

- A node is a separator if it does not contain any child nodes, text, or links.
- If an `<img>` node is regarded as a separator, its width, height, or its aspect ratio should be smaller than some threshold.
- `<applet>`, `<embed>`, `<object>`, `<frame>`, `<frameset>`, `<iframe>`, `<form>`, `<button>`, `<input>`, `<select>` and `<textarea>` nodes are not regarded as separators even if they do not contain any child nodes, text, and links.

We use the detected separators as indications of decomposition termination, just like [3][4]. The third step of salient block decomposition is a top-down traversal through the DOM tree to identify salient blocks. A node is regarded as a salient block only if it contains no separator in its offspring and there is at least one separator in the offspring of its parent. Otherwise, if its parent does not contain any separators in its offspring, the parent can be considered as a salient block. After traversal, the original web page is decomposed into a set of salient blocks, which is regarded as the compact visual representation of a web page.

### 3.2 Block Clustering

The granularity of the decomposed salient blocks is still too small and some of them still belong to the same topic. It is necessary for us to further group them into “topic blocks” according to their relevance. We use a hierarchical clustering algorithm to find these topic blocks. Topic block is regarded as the smallest unit of user’s interest in our approach.

The distance function in our approach calculates relevance between two blocks from two major aspects, i.e., adjacency and appearance. The neighborhood relationship and the minimum offset on X or Y-axis between blocks is used to calculate the distance in adjacency. The difference in background, foreground, font, size and alignment is detected to calculate the distance in appearance. The value from each aspect is multiplied by an importance factor and then summed up to get the final result.

With the definition of distance function, we apply a hierarchical clustering algorithm to find the topic blocks. Initially, each salient block is regarded as a cluster. We calculate the distance between two clusters  $\Phi_1$  and  $\Phi_2$  as follows.

$$DIS(\Phi_1, \Phi_2) = \min(Dis(A, B)) \quad \forall A \in \Phi_1, \forall B \in \Phi_2$$

According to the distance function value between each pair of clusters, we select the closest pair of clusters from them and merge them together. This process is repeated until distance between any two clusters is larger than a threshold. The remaining clusters are the topic blocks which represent the layout structure of the web page.

## 4 Layout Analysis

After page segmentation, we have obtained a set of topic blocks, which represent the layout structure of a web page. However, the content on the web page may change



daily and the layout of it may also change accordingly. In order to track the change of content of each topic, it is necessary for us to analyze the layouts of the web page in consecutive days and match blocks between them. For example, if a block, which contains links of sport news, on the web page of yesterday is identified as block #5, it should also be identified as block #5 today on the same page even if the links inside it are changed.

In order to match blocks between two web pages, it is necessary for us to find a formal representation to describe relationship among them. In this paper, we propose the Neighborhood Relationship Model, which is introduced in Section 4.1, to convert the topic blocks on a web page into a graph. Based on this model, we match the graph representations of two web pages to get the best matching of topic blocks. The layout matching algorithm will be presented in Section 4.2.

#### 4.1 Neighborhood Relationship Model

In the Neighborhood Relationship Model, the topic blocks on a web page are regarded as the vertices in a graph and the neighborhood relationship among them are regarded as the edges among those vertices. If block A is a neighbor of block B, they should satisfy the following constraints:

- A and B should not intersect with each other.
- There are no other blocks between A and B.
- The projection of A and B on X or Y axis should overlap.

There are two kinds of edges in this model, i.e., vertical edge and horizontal edge. If A is a top or bottom neighbor of B, there will be a vertical edge between vertex A and vertex B. If A is a left or right neighbor of B, there will be a horizontal edge between vertex A and vertex B.

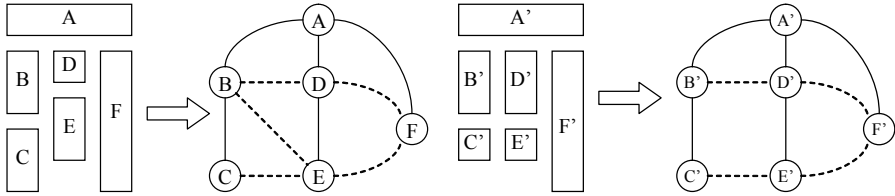
The Neighborhood Relationship Model is suitable to describe local structure of a web page. In most cases, it is relatively stable. Change in size of a block only affects a small part of the graph and some slight changes even have no effect on the structure of the graph.

#### 4.2 Layout Matching Algorithm

With the Neighborhood Relationship Model, the topic blocks on a web page are converted into a graph. However, because of the change of the amount of content, the size of blocks will change accordingly and thus affect the layout and graph representation of the web page. Figure 1 illustrates the change of layout and the corresponding change in the graph, in which the solid lines in the graph represent vertical edges and the dashed lines in the graph represent horizontal edges. In Figure 1, the blocks before the change are labeled as A, B, C, E, and F, and the blocks after the change are A', B', C', E', and F', correspondingly. As we can see, the changes of blocks C, D, and E causes change in the graph representation. Actually, only the change of block D affects the structure of the graph. Changes of block C and E have no effect on the

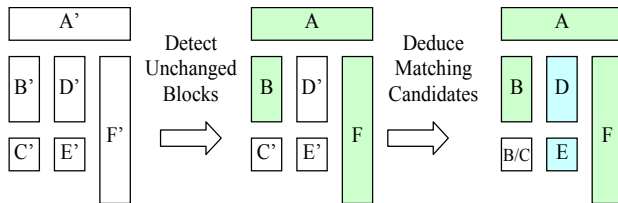
graph. This is the advantage of the Neighborhood Relation Model mentioned above. The remaining problem is how to match vertices between these two graphs.

Intuitively, because we analyze the layouts of a web page in consecutive days, there should be some blocks remaining unchanged on the page. Therefore, the first step of our layout matching algorithm is to match those unchanged blocks between the two pages. In this step, blocks in the two pages are matched according to their content (text or HTML inside a block). If block A on page P1 and block B on page P2 can be matched together, the content of A should be identical to the content of B and they should be unique in P1 and P2, respectively.



**Fig. 1.** Change of layout and graph representation (Solid lines represent vertical edges and dashed lines represent horizontal edges)

After matching of unchanged blocks, the second step of our layout matching algorithm is to deduce matching candidates for each unmatched block according to the neighborhood relationship among blocks. For example, in Figure 1, block A', B' and F' are matched with block A, B and F, respectively, in the first step because they are not changed. Because D is the first left neighbor of F and D' is also the first left neighbor of F', we can deduce that D is a candidate of D'. We do such deduction repeatedly on the graph until no new candidates can be found. Finally, as illustrated in Figure 2, blocks with unique candidates are marked as matched.



**Fig. 2.** Deduce matching candidates of blocks

The number of unmatched blocks is relatively small after the above two steps. The third step of our layout matching algorithm is to enumerate all possible matching plans for the remaining unmatched blocks according to their candidates and select the best one which matches maximal number of edges between two graphs. Two edges in two graphs are matched only if the vertexes of them are matched and the types of them are the same.

The nature of the third step of our layout matching algorithm is a graph matching problem, which has been proven to be NP-complete in [7]. Due to the deduction

process in the second step, the number of candidates of each unmatched block is greatly reduced. According to the experiment results (refer to Section 6), the number of unmatched blocks is usually smaller than 15. Therefore, the enumeration algorithm in the third step can find the best matching plan within one minute in most cases.

## 5 Interest Detection

In this section, we will present the design of our personalized information agent and the corresponding method to detect user's interest. Our personalized information agent consists of three modules, i.e., Tracking Module, Analysis Module and Retrieval Module.

In the Tracking Module, we aim to record two kinds of information to create user profile. The first is the browsing behavior of the user, mainly his click actions on web pages. The second is the layout of web pages browsed by the user. These pages are first split into topic blocks by the page segmentation algorithm introduced in Section 3. Later, if a web page has been visited before, topic blocks on it will be matched to old ones by the layout matching algorithm introduced in Section 4.

The Analysis Module is the core of our personalized information agent. It detects user-interested blocks in two steps. In the first step, we select the user-interested web pages according to the frequency at which the user browses these pages. The frequency is calculated using the following equation.

$$Frequency(Page) = \frac{VisitingCount(Page)}{Time_{Now} - Time_{StartLog}} \quad (1)$$

According to the frequency, we select top 40 pages that are the most frequently browsed and then analyze the user's interest on them. In the second step, we also use the frequency to estimate the user's interest in a topic block using the following equation.

$$Frequency(Block) = \frac{\sum_{P_i \in BrowsedFrom(Block)} VisitingCount(P_i)}{Time_{Now} - Time_{StartLog}} \quad (2)$$

For each page, we select top 20 frequently clicked blocks as candidates of user-interested blocks. The ID numbers of these blocks and their corresponding URLs are passed to the Retrieval Module to extract newly updated content in these blocks.

In the Retrieval Module, our agent system checks the user-interested pages regularly. The content in these blocks is extracted and compared to the old ones in the user's database. Changed content is picked up and presented to the user in a convenient way. In our agent system, the changed content is reorganized into a web page which is set as the startup page of the user's browser.

## 6 Experiment

In order to evaluate our approach, we have tracked the home pages of 12 popular web sites over 1 month. For each web site, we have collected 25 copies of its home page

in consecutive days. Totally, we have obtained 300 pages as our testing data sets. First, we use our page segmentation algorithm to split these web pages into topic blocks. Based on these topic blocks, we match the layouts of a web page in consecutive days by the first and second steps of our layout matching algorithm. Table 1 illustrates average amounts of blocks segmented from the home pages of those 12 web sites and the corresponding average number of unmatched blocks on each page after the first two steps of layout matching. We can see that the number of unmatched blocks is relatively small after the first two steps, which guarantee the efficiency of the third step of layout matching in most cases. Moreover, it also proves that the topic blocks, which are segmented from the web page by our page segmentation algorithm, is a relatively stable representation of the web and our page segmentation algorithm is therefore applicable to this kind of problems.

**Table 1.** Unmatched blocks after the first two steps in the layout matching algorithm

Web Site	Avg. Unmatched/Total	Web Site	Avg. Unmatched/Total
cnet.com	8.3/114.4	cnn.com	7.8/39.1
excite.com	0.8/49.0	flowgo.com	0.2/23.2
go.com	2.1/58.9	hjsm.net	1.1/37.9
nbc.com	0.5/15.9	netscape.com	6.9/43.9
sina.com.cn	8.7/107.5	travelocity.com	0.04/31.2
windowsmedia	1.8/27.4	yahoo.com	9.7/48.8

Based on the above matching result, we match the remaining blocks by the third step of our layout matching algorithm. Table 2 illustrates the distribution of the numbers of unmatched blocks between two web pages in consecutive days before the third step and the corresponding executing time of our layout matching algorithm with the third step. Our testing platform is a PC with a P4 1.8G CPU and 256MB RAM. For those pages, which have more than 15 unmatched blocks, we think there must be some significant changes in the layout of the page and just regard them as failures. From Table 2 we can see that the enumeration algorithm in the third step of our layout matching algorithm is able to find the best matching plan within one minute under most situations. Although one minute is too slow for real-time response, since our agent system can work off-line and does not need to give response in real time, the efficiency of our layout matching algorithm is acceptable. Moreover, in most cases (nearly 90%), our algorithm can even respond in real time.

**Table 2.** Distribution of the number of unmatched blocks and executing time of the layout matching algorithm

Number of block	Percent	Executing Time
0~9	89.4%	<170ms
10~15	7.8%	<60s
>15	2.8%	-

**Table 3.** Error rate of block matching

Web Site	Error	Web Site	Error	Web Site	Error
go.com	1.3%	hjsm.net	0.3%	cnet.com	5.7%
nbc.com	1.0%	netscape.com	6.8%	excite.com	1.1%
sina.com.cn	6.8%	travelocity.com	0.1%	Cnn.com	11.5%
windowsmedia.com	3.8%	yahoo.com	6.7%	flowgo.com	0.2%

After the blocks of the pages in consecutive days are matched together, we perform a manual check on the matching plan. Those blocks, which are not matched or matched incorrectly, are regarded as errors. Table 3 illustrates the error rate of block matching in all 12 web sites. Most of the errors are caused by significant changes in layouts. However, we can see that the accuracy of our layout matching algorithm for most web sites is satisfactory.

We run a simulation on the testing sets in order to evaluate the accuracy of our approach to detecting user's interest. A user is invited to browse each home page and click on the links in which he is interested to create the user profile. Based on this profile and equations (1) and (2) in Section 5, user-interested blocks are picked out and the updated content in them is extracted by our personalized information agent. The updated content of each day is reorganized into a page of interest. The user is also asked to browse those pages and perform a manual check on them. Two situations are regarded as failures. The first is that the user-interested blocks cannot be found. The second is that the user is not interested in the blocks that we find. Table 4 illustrates corresponding evaluation results on those 12 web sites. As we can see, our algorithm achieves satisfactory accuracy in most sites. The worst case is the www.sina.com.cn. The accuracy for it is relatively low because the position of user-interested blocks changes every the other day. We are going to solve this problem by introducing some semantic constraints in our further work.

**Table 4.** Accuracy of user interest detection

Web Site	Correct	Web Site	Correct	Web Site	Correct
go.com	95.8%	hjsm.net	100%	cnet.com	97.5%
nbc.com	83.3%	netscape.com	100%	excite.com	100%
sina.com.cn	56.9%	travelocity.com	100%	cnn.com	84.7%
windowsmedia.com	95.8%	yahoo.com	81.9%	flowgo.com	97.9%

A preliminary user evaluation has been done on our personalized information agent. First, the interests of the user, who performs in the above experiments in all 12 web sites, are concluded according to his feedback. After that, we invite another 3 users, who have similar interests and are also familiar with those web sites, to browse those web sites and look for the content in which the first user is interested. These 3 users' browsing time on these web sites is recorded and the average time is listed in Table 5.

**Table 5.** User evaluation

Web Site	Avg. Time	Web Site	Avg. Time
cnet.com	37s	cnn.com	60s
excite.com	42s	flowgo.com	21s
go.com	54s	hjsm.net	26s
nbc.com	34s	netscape.com	52s
sina.com.cn	59s	travelocity.com	33s
windowsmedia.com	38s	yahoo.com	26s
Our Generated Page	274s		

These 3 users are also asked to browse the pages generated by our agent system to find the same content. The average browsing time on our pages is also listed in Table 5 (the last row). Assume the average browsing time on web site  $i$  is  $t_i$  (listed in Table 5), the detection accuracy of user's interest on web site  $i$  is  $p_i$  (listed in Table 4), and the average browsing time on our pages is  $T$  (listed in Table 5). We evaluate the benefit of our agent system by the following equation:

$$Benefit(T) = 1 - \frac{T}{\sum_{i=1}^n t_i \times p_i} \quad (3)$$

The benefit of our agent system in this experiment is 36.5% over all the 12 web sites, which means we can save 36.5% browsing time on these pages for the user. Moreover, all of the 4 users in our experiments think this kind of information agent can help them find their interested information quickly and therefore improves their browsing experience.

## 7 Conclusions

In this paper, we propose a novel approach to detecting user's interests on web pages and develop a personalized information agent based on it. The key feature of our approach is that we introduce a vision-based approach to this problem so that user's interests can be detected in smaller granularity than existing approaches and hence with better accuracy. The experiments on 12 web sites over 1 month show that our approach has achieved satisfactory efficiency and accuracy. Moreover, the preliminary user evaluation has shown that our personalized information agent significantly saves user's browsing time and improves user's browsing experience.

## Acknowledgement

The work described in this paper was fully supported by a grant from City University of Hong Kong (Project No. 7001644).

## References

1. Andrew, J.K., and Javed, M.: Topic Detection and Interest Tracking in a Dynamic Online News Source. The 3rd ACM/IEEE-CS Joint Conference on Digital Libraries, 2003.
2. Buttler, D., Liu, L., and Pu, C.: A Fully Automated Object Extraction System for the World Wide Web. International Conference on Distributed Computing Systems, 2001.
3. Chen, Y., Ma, W.Y., and Zhang, H.J.: Detecting web page structure for adaptive viewing on small form factor devices. The 12th International Conference on World Wide Web, 2003.
4. Gu, X.D., Chen, J.L., Ma, W.Y., and Chen, G.L.: Visual Based Content Understanding towards Web Adaptation. The 2nd International Conference on Adaptive Hypermedia and Adaptive Web-based Systems, 2002.
5. Hyoungh, R.K., and Philip, K.C.: Learning Implicit User Interest Hierarchy for Context in Personalization. International Conference on Intelligent User Interfaces, 2003.
6. Lauren, W.: Document Object Model Specification. <http://www.w3.org/DOM/>.
7. Mehlhorn, K.: Graph Algorithm and NP-Completeness. Berlin, Heidelberg, Springer-Verlag, 1984.
8. Nanno, T., Saito, S., and Okumura, M.: Structuring Web pages based on Repetition of Elements. The 7th International Conference on Document Analysis and Recognition, 2003.

# Improved Link-Based Algorithms for Ranking Web Pages\*

Ziyang Wang

Department of Computer Science, New York University, New York, NY 10012  
ziyang@cs.nyu.edu

**Abstract.** Several link-based algorithms, such as PageRank [7], HITS [4] and SALSA [5], have been developed to evaluate the popularity of web pages. These algorithms can be interpreted as computing the steady-state distribution of various Markov processes over web pages. The PageRank and HITS algorithms tend to over-rank tightly interlinked collections of pages, such as well-organized message boards. We show that this effect can be alleviated using a number of modifications to the underlying Markov process. Specifically, rather than weight all outlinks from a given page equally, greater weight is given to links between pages that are, in other respects, further off in the web, and less weight is given to links between pages that are nearby. We have experimented with a number of variants of this idea, using a number of different measures of “distance” in the Web, and a number of different weighting schemes. We show that these revised algorithms often do avoid the over-ranking problem and give better overall rankings.

## 1 Introduction

The ranking of Web pages returned in response to a user query combines a measure of the relevance of the page to the query together with a query-independent measure of the quality of the page. The latter measure is based on the structure of the Web, considered as a directed graph of pages and links: A page with many in-links is presumed to be a high-quality page, particularly if (circularly) the links come from pages that are themselves high-quality. A number of techniques have been developed to rank Web pages based purely on the structure of hyperlinks. The best known of these are the PageRank algorithm [7], used in Google; the HITS algorithm [4], proposed by Kleinberg; and the SALSA algorithm [5]. (The original proposal for HITS and SALSA is to apply them to a collection of pages found to be relevant to a query, but the same algorithms can be applied to any collection of web pages.) These and similar algorithms can be viewed as computing the steady-state distribution of various Markov processes over Web pages. Algorithms of this kind tend to over-rank tightly

---

\* This work is supported by NSF grant #IIS-0097537. There is an extended version of this paper as NYU computer science dept. technical report TR2003-846, online at <http://csdocs.cs.nyu.edu/Dienst/UI/2.0/Describe/ncstrl.nyu.cs/TR2003-846>.



interlinked collections of pages, such as message boards. Intuitively, a Markov process executing a random walk through the Web tends to get stuck inside such a collection, crossing links from one of the pages to another. However, the large number of links between these pages does not constitute any independent endorsement of quality, or at best is a comparatively weak endorsement of quality. We call this the “Circular Contribution effect”; effectively, each of the pages in such a collection boosts the evaluation of all the others, and hence, indirectly, of itself. To counteract this undesirable effect, we propose to modify the underlying Markov processes by giving different weights to different outlinks from a page. A link from  $P$  to  $Q_1$  is weighted as more important than a link from  $P$  to  $Q_2$  if, in some sense,  $Q_1$  is “further” than  $Q_2$  from  $P$  in the Web. We present three different definitions of this notion of “distance in the Web” and a number of different weighting schemes. Adapting each of these modifications to the PageRank and HITS algorithms, we show experimentally that the revised algorithm is much less prone to the circular contribution effect.

### 1.1 Major Link-Based Ranking Systems and Their Properties

We begin by discussing the three algorithms PageRank [7], HITS [4], and SALSA [5] and some of their properties. The detailed algorithms will be presented in Section 4.1. The stochastic model used in PageRank is a random walk through the web. In its original model, the web surfer follows the out link from his current page, and jumps to linked pages with equal probabilities; with small probability (0.15) he jumps randomly in the web. The PageRank is the steady-state distribution of this stochastic process. HITS and SALSA are based on a model of the web that distinguishes hubs and authorities. Each page is assigned a “hub” value and an “authority” value. The hub value of page  $H$  is a function of the authority values of the outlinks from  $H$ ; the authority value of page  $A$  is a function of the hub values of the inlinks into  $A$ . Borodin et al. [2] show that this can be interpreted in terms of a stochastic process that alternates traveling forward and backward across links. In the computational models of these stochastic processes, if there exists a path from node  $i$  to node  $j$  and also a path from node  $j$  to node  $i$  in the selected web graph, then the rank value of node  $i$  will boost the rank value of node  $j$  and vice versa; indirectly, each node ends up “endorsing” itself. We call this effect *Circular Contribution*. In general, Circular Contribution tends to be a feature of stochastic models, and often a useful one. But it can also lead to incorrect results. In general, if there is a Web Local Aggregation in the web graph, discussed in the next section, the Circular Contribution effect can mislead these algorithms into inaccurate rankings.

### 1.2 Web Local Aggregation Effect in Stochastic Ranking Systems

A hyperlink implies some relationship between the linked documents. We divide hyperlinks into two major categories based on their functionalities:

Category 1: Links are used for information reference and information association. For example, students may add university home page to their personal home pages

because the university home page is an important information source for university community. Such links works as information reference based on human knowledge. Topic-related pages may link to each other as information association.

Category 2: Links are used for directories. This makes web browsing meaningful. Directory links frequently provide weak information reference and association. A discussion board may add links to many messages. The topics of messages may be very diverse and there may be even no association between father page and child pages.

When the hyperlinks functioning as directories gives weak information reference and association, they may cause negative effect in global web ranking. It is very common that the father page hosting many directory links may gain many back links from its children. Thus the father page and child pages form cycles and the Circular Contribution effect happens. In ranking systems that use stochastic models, all pages in such collections will have their ranking raised as a result. We call this effect *Web Local Aggregation*. A Web Local Aggregation is characterized by the property that the number of intra-links within the collection is very large and is significantly larger than those of the in-bound links entering the collection and out-bound links leaving the collection. When information reference and association of the hyperlink is weak in such situation, the ranking biased by Web Local Aggregation does not correctly evaluate the global importance of web pages.

### 1.3 Our Work

To overcome the negative effect of local aggregation and improve the effectiveness of ranking systems, we propose the ideas of *Hyperlink Evaluation and Evaluation-based Web Ranking*.

In the stochastic models discussed in section 1.1, all the outlinks on a certain page or all the inlinks to a certain page are evaluated equally. The modifications that we propose, by contrast, give different weights to different links, so as to avoid or alleviate the effect of Web Local Aggregation. Based on link evaluation and the frameworks of existing stochastic web ranking algorithms, new ranking algorithms are proposed which can alleviate the negative effect of Web Local Aggregation effectively.

The remainder of this paper is organized as follows: Section 2 presents three different methods for hyperlink evaluation: *collection-interlink amplification*, *collection-rank-based evaluation*, and *minimal back-distance based evaluation*. Section 3 discusses how the PageRank, HITS, and SALSA algorithms can be modified to use the methods of hyperlink evaluation. Section 4 presents several experiments on New York University web site and analyzes the effectiveness of the new algorithms. Due to space limit, more discussions and results are presented in the extended version only [8].

## 2 Metrics of Hyperlink Evaluation

We propose two general methods to reduce the effect on rankings of Web Local Aggregation. The first method is to weight cross-links between different domains more strongly than links between pages in a single domain. The second general method is to weight the link between two pages more strongly as a function of the length of the shortest path in the Web in the reverse direction of them. This directly attacks the Circular Contribution effect, as the contribution of short cycles in the Web to the evaluation of web pages is reduced. Based on these two methods, we present three metrics of hyperlink evaluation here: collection-interlink amplification, collection-rank-based evaluation and minimal back-distance based evaluation.

### 2.1 Metric 1: Collection-Interlink Amplification

Our analysis clusters the web into collections using host information. That is, two pages are placed in the same collection if and only if they belong to the same host. This metric evaluates cross-links between different collections greater than links within the collections by a fixed ratio. Mathematically, denote  $v_{ij}$  as the value of link  $i \rightarrow j$ ,  $Coll(i)$  as the collection index of page  $i$ . Let  $m_i$  be the number of links from page  $i$  within the collection and  $n_i$  be the number of links from page  $i$  that leave the collection. Let  $C_i = \delta m_i + n_i$ . The value of link  $(i, j)$  is given as

$$v_{ij} = \begin{cases} \delta / C_i & \text{if } Coll(i) = Coll(j) \\ 1 / C_i & \text{otherwise} \end{cases} \quad (1)$$

where  $\delta$  is a constant within  $[0, 1]$ . The coefficient  $C_i$  normalizes the values  $v_{ij}$  such that the sum of  $v_{ij}$  is 1 respect to index  $j$ . The value  $1/\delta$  gives the ratio how cross-links between collections are evaluated greater than local links. By giving more evaluation on cross links, the surfer can obtain higher probability to jump out of web local aggregation that frequently happens within a collection.

### 2.2 Metric 2: Collection-Rank-Based Evaluation

The second metric is also devised using web collection information but an improvement of Metric 1. We consider not only the cross link between collections, but also the reliability of the destination host of a link. We use a two-stage evaluation approach to evaluate the importance of pages in the web graph: the collections are evaluated first then hyperlink evaluation can be improved using collection evaluation. We use a PageRank-like algorithm to evaluate collections. Suppose there are  $q$  different collections in the web graph and let  $N_{ij}^{Coll}$  be the number of outer links from collection  $i$  to collection  $j$ . Then the transition matrix  $T^{Coll}$  is

$$T^{Coll} = \varepsilon U^{Coll} + (1 - \varepsilon) M^{Coll} \quad (2)$$

where  $U^{Coll}$ ,  $U_{ij}^{Coll} = 1/q$ , and  $M^{Coll} : M_{ij}^{Coll} = \frac{N_{ij}^{Coll}}{\sum_k N_{ik}^{Coll}}$ .

The ranking value vector of collections, denoted as  $V^{Coll}$ , is the stationary distribution of transition matrix  $T^{Coll}$ . When the collection evaluation  $V^{Coll}$  is given, we can define the hyperlink evaluation as:

$$v_{ij} = \begin{cases} \frac{\delta \times S(V_{Coll(j)}^{Coll})}{C_i} & \text{if } Coll(i) = Coll(j) \\ \frac{S(V_{Coll(j)}^{Coll})}{C_i} & \text{otherwise} \end{cases} \quad (3)$$

where  $C_i$  is the normalization coefficient such that the sum of  $v_{ij}$  is 1 respect to index  $j$ , and  $S: R \rightarrow R$  is an increasing function to scale the results of collection evaluation. This link evaluation corresponds to the following surfer behavior in a random walk: during the walk, the surfer can “see” the collection information of its neighbors. When he makes the transition selection, he prefers to jump to the page whose collection is more trustable.

### 2.3 Metric 3: Minimal Back-Distance (MinBD) Based Evaluation

This metric directly identifies the cycle information in the web graph and evaluates the links accordingly. In Circular Contribution, we know mutual contribution of two nodes is affected by the length of the path between them. The smaller the length is, the greater the contribution is. Thus, for each link  $i \rightarrow j$ , the backward contribution of node  $j$  to node  $i$  is dominated by the minimal length of all possible paths from node  $j$  to node  $i$  in the web graph. We call this length minimal back-distance (MinBD). Denote  $MinBD_{ij}$  as the minimal back distance from  $j$  to  $i$ . For each link  $i \rightarrow j$  the mathematical representation of this metric is

$$v_{ij} = \begin{cases} \frac{f(\Omega)}{C_i} & \text{if } MinBD_{ij} \geq \Omega \\ \frac{f(MinBD_{ij})}{C_i} & \text{otherwise} \end{cases} \quad (4)$$

where  $\Omega$  is a threshold,  $C_i$  is the normalization coefficient such that the sum of  $v_{ij}$  is 1 respect to index  $j$ ,  $f: R \rightarrow R$  is an increasing function satisfying  $f(0) \geq 0$ . The threshold  $\Omega$  is set as 30, which can fit the current size of web diameter quite well [1, 3]. Let's consider the random walk guided by this link evaluation. Suppose the web surfer can “see” further than the neighbors at any page. An effective web walk strategy should explore the web as much as possible and not go back to the visited local pages frequently. As those links with small MinBD are evaluated less, the Circular Contribution effect is reduced, which can effectively alleviate the negative effect of

Web Local Aggregation whose web graph contains many short cycles. MinBD can be efficiently computed with a DFS-based graph algorithm. The detailed discussion of this algorithm has been presented in the extended version of this paper [8].

### 3 Modify Stochastic Process Using Hyperlink Evaluation

Supported by the three metrics of hyperlink evaluation, we propose a new ranking strategy named *Evaluation-based Web Ranking*. The basic idea here is that by applying link evaluation to the stochastic ranking algorithms, the new algorithms can improve ranking effectiveness compared with the original ones. We first review the mathematical frameworks of three different algorithms here: PageRank [7], HITS [4] and SALSA [5]. Then we discuss how to apply hyperlink evaluation into these ranking frameworks to produce new improved algorithms. The improved algorithms are independent with what actual link evaluation metric is used. For the mathematical representation of SALSA and the improved version, refer to [8].

#### 3.1 The Mathematical Framework of Three Existing Ranking Algorithms

**PageRank Algorithm.** A web graph of  $n$  pages can be represented using an  $n$ -by- $n$  adjacency matrix  $A$ , where  $A(i, j)$ -element is 1 if page  $i$  links to page  $j$ , and 0 otherwise. The PageRank algorithm first constructs a probability transition matrix  $M$  by normalizing each row of adjacency matrix  $A$  to sum to 1. The idea of this algorithm is inferred from random walk within a graph. When a person is at page  $i$ , with probability  $(1-\epsilon)$ , it uniformly picks a URL link from this page and transits to the target page of this link. With probability  $\epsilon$ , it jumps to any other page in the web with uniform probability. The final transition matrix  $T$  is

$$T = \epsilon U + (1-\epsilon)M \quad (5)$$

where  $U$  is an  $n$ -by- $n$  matrix of uniform transition probabilities having  $U_{ij} = 1/n$  for all  $1 \leq i, j \leq n$ . The vector of PageRank scores  $p$  is then defined to be the stationary distribution of this Markov chain. The stationary distribution is the eigenvector of the transition matrix and satisfies

$$(\epsilon U + (1-\epsilon)M)^T p = p \quad (6)$$

**HITS Algorithm.** The HITS algorithm points that a page has high “authority” weight if it is linked to by many pages with high “hub” weight, and that a page has high hub weight if it links to many authoritative pages. Given a set of  $n$  web pages, the HITS algorithm firstly forms the  $n$ -by- $n$  adjacency matrix  $A$ , whose  $(i, j)$ -element is 1 if page  $i$  links to page  $j$ , and 0 otherwise. It iterates the following equations:

$$a^{t+1} = A^T \cdot h^t \quad (7)$$

$$h^{t+1} = A \cdot a^t \quad (8)$$

where  $a$  and  $h$  are the vectors of authority values and hub values. For each iteration, a normalization step is then applied, so that the vectors  $a$  and  $h$  become unit vectors in some norm. Kleinberg [4] proves that for a sufficient number of iterations the vectors  $a$  and  $h$  converge to the principle eigenvectors of the matrices  $A^T A$  and  $AA^T$ . A more stable HITS is given by Ng et al. [6]. It introduces a small uniform jump probability to any other page for each transition as PageRank algorithm does.

### 3.2 PageRank-Based Algorithm Using Link Evaluation

In PageRank algorithm, the small coefficient of uniform jumping probability,  $\varepsilon$ , counts for an important factor of PageRank algorithm [6] and can be used to bias PageRank value distribution. We propose two levels of Evaluation-based Web Ranking for PageRank-like algorithm: the first level of evaluation is that the PageRank transition probabilities can be evaluated by link evaluation; the second level of evaluation is that the uniform jumping probability  $\varepsilon$  for each page can be evaluated by the link evaluation on that page. These findings lead to the discovery of two improved algorithms in the mathematical framework of PageRank algorithm: *fixed  $\varepsilon$  algorithm* and evaluation-based *non-fixed  $\varepsilon$  algorithm*.

**Level 1: Fixed  $\varepsilon$  algorithm.** The original PageRank transition matrix  $M$  in Section 3.1.1 is modified as

$$\tilde{M} : \tilde{M}_{ij} = v_{ij} \quad (9)$$

where  $v_{ij}$  is any one of the link evaluations given in Section 2. Since the link values have been normalized,  $\tilde{M}$  is a well-formed transition matrix. Clearly, we replace the original matrix  $M$  with  $\tilde{M}$  in Section 3.1.1 and apply the fixed uniform jumping probability to form the final transition as Eq. (6) does.

**Level 2: Evaluation-based non-fixed  $\varepsilon$  algorithm** The hyperlink evaluation is the first level evaluation in reducing the local aggregation of the web. In the second level, we evaluate the overall quality of all out links from a page, and then decide how to choose  $\varepsilon$ . For any page  $i$ , we calculate the average of hyperlink values of all out links on page  $i$ . Because the normalization of hyperlink evaluations in Section 3 is performed on different bases for pages with different number of out links, we use the non-normalized evaluation to calculate average quality of links on a single page. The average quality of the out links on page  $i$  is defined as

$$AvgQual(i) = \frac{\sum_{j:i \rightarrow j} V_{ij}}{N_i} \quad (10)$$

where  $V_{ij} = v_{ij} \times C_i$  is the un-normalized link value and  $N_i$  is the number of outlinks on page  $i$ . The uniform jumping probability for page  $i$ ,  $\varepsilon_i$ , is then evaluated as

$$\varepsilon_i = \varepsilon_{base} + \varepsilon_{adjust} \times g(AvgQual(i)) \quad (11)$$

where  $\varepsilon_{base}$  is the global base and  $\varepsilon_{adjust}$  is the adjustment range.  $g()$  is an decreasing function and maps  $AvgQual(i)$  into  $[0, 1]$ . For example, if  $AvgQual(i)$  is in  $[1, \infty]$ ,  $g(x)=1/x$  satisfies. The greater value  $AvgQual(i)$  has, the smaller probability it will jump to an arbitrary page.

Let  $p^t$  be the ranking vector after iteration  $t$ . Iteratively computing rank is shown as follows

$$p_i^{t+1} = \sum_j \frac{\varepsilon_j \times p_j^t}{n} + \sum_{j:j \rightarrow i} (1 - \varepsilon_j) \times v_{ij} \times p_j^t \quad (12)$$

where  $n$  is the total number of pages in the web graph. To reduce complexity, the first term must be calculated only once for each iteration. And after each iteration, vector  $p^t$  must be normalized.

### 3.3 HITS-Based Algorithm Using Link Evaluation

To modify stochastic process in HITS ranking system, we can apply the link values as the weights to HITS algorithm. The non-zero entries of new transition matrix  $\tilde{A}$  is defined as

$$\tilde{A} : \tilde{A} = \frac{v_{ij}}{\max_{k:i \rightarrow k} v_{ik}} \quad (13)$$

where  $i \rightarrow j$  is a link in the web graph. All entries of  $\tilde{A}$  are within  $[0,1]$  in this mathematical representation. Replacing  $A^T$  and  $A$  in HITS algorithm with  $(\tilde{A})^T$  and  $\tilde{A}$ , we get our improved HITS-based algorithm.

## 4 Experiments

Our experiment data set is built on a crawl on the entire web of New York University in October, 2002. The crawler narrows its downloading to the URLs with host name ending with “nyu.edu”. After removing repetitions of web pages and non-text pages, the total number of URLs is 98,349. These URLs make up a web graph consisting 723,380 hyperlinks. The ratio between number of hyperlinks and URLs is 7.4. The study of the in-link and out-link distributions shows they obeys power law with power coefficients of 1.94 and 2.24. These properties are in very good agreement with the study on the global Web [3]. Basically, PageRank-based algorithms are run on different data sets as HITS-based or SALSA-based algorithms. For PageRank-based algorithms, they can be applies directly on a well linked web graph. For HITS-based or SALSA-base algorithms, they are generally applied on topic related data sets. But in Section 1.1 and Section 3.1.1, we have known that their computation models are independent with data set. Therefore, we apply HITS-base and SALSA-based algorithms directly on our data set for analyzing purpose. To make ranking unique in each of such ranking systems, we choose *authority* ranking as the evaluation of im-

portance of web pages. We have found that Web Local Aggregation in the NYU web does indeed have a negative effect on the rankings output by the standard algorithms. Table 1 shows the top 15 pages given by PageRank algorithm by setting  $\varepsilon = 0.15$  in Eq. (6). The pages ranked from 9 to 12 are the directory pages of a message board archive in the medical school of NYU. Each of the directory pages hosts the links to the same collection of 958 message pages of different orders. And each message page has links to the four directory pages and links to the previous and following messages. These directory pages are ranked unexpectedly high in the whole collection of about 100,000 pages, as compared to their actual significance, intuitively judged. However, since there are many short cycles in this collection, the Circular Contribution effect is significant and produces this inaccurate ranking distribution. Table 1 also shows the selected top ranked pages using stable HITS algorithm. A local collection describing a project called “Proteus” takes top 25 positions higher than the portal page of NYU.

**Table 1.** Selected rank of PageRank (left) and stable HITS (right)

Rank	URL	Rank	URL
1	www.nyu.edu	1	apple.cs.nyu.edu/proteus/local/data/OnlineProc /ACL02/MAIN/index.htm
2	www.nyu.edu/bin/phfnyu	2	apple.cs.nyu.edu/proteus/local/data/OnlineProc/ACL02/EMNLP/index.htm
3	www.nyu.edu/library/bobst	3	apple.cs.nyu.edu/proteus/local/data/OnlineProc/ACL02/S2S/index.htm
4	www.stern.nyu.edu/acc/about	4	apple.cs.nyu.edu/proteus/local/data/OnlineProc/ACL02/DIALOG/index.htm
5	www.nyu.edu/gsas	5	apple.cs.nyu.edu/proteus/local/data/OnlineProc/ACL02/DEMOS/index.htm
6	www.law.nyu.edu		.....
7	www.med.nyu.edu	13	www.nyu.edu
8	monod.biomath.nyu.edu/index/papers.html	14	apple.cs.nyu.edu/proteus/local/data/OnlineProc /ACL02/WSD/contents.htm
9	mchip00.med.nyu.edu/.../date.html	15	www.nyu.edu/bin/phfnyu
10	mchip00.med.nyu.edu/.../index.html		
11	mchip00.med.nyu.edu/.../author.html		
12	mchip00.med.nyu.edu/.../subject.html		
13	www.law.nyu.edu/index.html		
14	rmm-java.stern.nyu.edu/.../index.html		
15	www.law.nyu.edu/search		

In support of our theoretical analysis in the previous sections, we implement the following algorithms: the original PageRank (PR), stable HITS (HT), and SALSA algorithms (SA); PageRank-based Level 1 evaluation algorithms using link evaluations of Metric 1 and Metric 2 (PRM1, PRM2); PageRank-based Level 1 and Level 2 evaluation algorithms using link evaluation of Metric 3 (PRL1M3, PRL2M3); HITS-based and SALSA-base algorithms using link evaluations of Metric 1, Metric 2 and Metric 3 (HTM1, HTM2, HTM3, SAM1, SAM2, SAM3).

Given those improved ranking algorithms, the greatest challenge is how to evaluate them compared with the original one. We use an *aggregation-tracing* method to evaluate the effectiveness for different ranking algorithms in avoiding negative effect of Web Local Aggregation: first, we manually identify several sub collections of local aggregation in our data set; second, we trace how their ranks change for different ranking systems. The manually selected sub collections of local aggregation are given in Table 2. Collection MCHIP and END98-01 have similar structure which contains four directory links hosting the same message archives. PROT is the collection taking



the top positions in the ranking computed by HITS algorithm. A common property of these collections is that the number of intra-links within the collections is very large, and the percentage of intra-links respect to the involved links is all more than 81.0%. The local structures of these collections comply with our description of Web Local Aggregation. In the following discussions, we'll see that the top rank of these collections are very high in the overall ranking given by three original algorithms as we predicted in Section 1. Tracing the rank change of these local aggregations will give us the clue how our algorithms can improve ranking quality.

**Table 2.** Humanly identified sub collections of Web Local Aggregations

Abbrev	URL prefix	Size	Intra-link	In	Out
MCHIP	mchip00.med.nyu.edu/lit-med/archives/messages/	962	12967	2	45
PROT	apple.cs.nyu.edu/proteus/local/data/OnlineProc/	875	7469	6	26
END98	endeavor.med.nyu.edu/pipermail/lit-med/1998/	193	2850	10	12
END99	endeavor.med.nyu.edu/pipermail/lit-med/1999/	255	3876	10	21
END00	endeavor.med.nyu.edu/pipermail/lit-med/2000/	223	3336	10	76
END01	endeavor.med.nyu.edu/pipermail/lit-med/2001/	345	5162	10	255
ARTG	www.cs.nyu.edu/artg/telecom/fall99/lecture notes/	504	4148	3	966

In measuring the change of different ranking, we use the average difference  $ADiff$  and difference of highest rank  $HDiff$  to evaluate the rank change of a sub collection between two different ranking distributions. Let  $\Phi$  be a sub collection of local aggregation,  $R_1(i)$  and  $R_2(i)$  be the ranks of page  $i \in \Phi$  in two different ranking distributions. The measures of  $ADiff$  and  $HDiff$  are defined as follows:

$$ADiff(\Phi, R_2, R_1) = \frac{\sum_{i \in \Phi} (R_2(i) - R_1(i))}{Size(\Phi)} \quad (14)$$

$$HDiff(\Phi, R_2, R_1) = \min_{i \in \Phi} R_2(i) - \min_{i \in \Phi} R_1(i) \quad (15)$$

Suppose the negative effect of Web Local Aggregation happens on collection  $\Phi$  in rank  $R_1$ . If the value of  $ADiff(\Phi, R_2, R_1)$  or  $HDiff(\Phi, R_2, R_1)$  is positive, it shows such negative effect has been alleviated by  $R_2$ , otherwise not. The larger such value is, the more effective the ranking  $R_2$  is. Table 3 and Table 4 (for results about SALSA, refer to [8]) give our main results. Analyzing the data in these tables, we draw the following conclusions as our major results of this paper:

*PageRank and HITS algorithms are very sensitive to Web Local Aggregation.* In PageRank and HITS algorithms, all selected collections except PROT have average rank within top 27% of the whole data set. The highest rank of MCHIP given by PageRank is 9 as we have pointed out in Table 1. The top rank of PROT given by HITS is 1 which is surprisingly bad as pointed out by Table 1. The top ranked pages of all selected collections, i.e. the Top(PR), Top(HT), Top(SA) in Table 3, 4, 5, are all ranked very high in the whole collection. All of them are in the top 496 pages and are within top 0.5% of the whole data set.

*PageRank and HITS algorithms give similar rank distribution.* The average ranks of the selected collections given by PageRank and HITS algorithms obey the same order: MCHIP < END98-01 < ARTG < PROT. For the four collections of END98-01, the average ranks are close to one another both in the ranking given by PageRank and HITS algorithms. The standard variations of them are 1401.8 and 491.5, and are within 9% and 3% of the cumulative average of Avg(PR) and Avg(HT). The average of  $|Avg(PR) - Avg(HT)|$  for selected collections is 5184.1, which counts for 5.3% of the size of total data set.

**Table 3.** Effectiveness of improved PageRank-based algorithms

Coll.	Avg(PR)	ADiff(PRM1,PR)	ADiff(PRM2,PR)	ADiff(PRL1M3,PR)	ADiff(PRL2M3,PR)
MCHIP	12015.7	1744.2	2084.8	5927.8	5743.2
PROT	59670.4	-1985.2	-2490.5	-8751.0	-8144.7
END98	14199.5	1963.6	2490.5	3768.1	3507.8
END99	14485.9	1986.4	2547.7	3854.7	3717.7
END00	15666.9	1700.9	2243.4	2543.1	2388.9
END01	17634.0	1459.0	1850.9	1863.7	1719.1
ARTG	25024.9	65.1	299.4	539.3	916.9
Coll.	Top(PR)	HDiff(PRM1,PR)	HDiff(PRM2,PR)	HDiff(PRL1M3,PR)	HDiff(PRL2M3,PR)
MCHIP	9	9	26	25	24
PROT	496	127	108	55	56
END98	139	67	82	59	59
END99	95	60	70	54	55
END00	129	55	66	42	40
END01	73	44	53	42	38
ARTG	47	6	7	20	21

*The ranking systems using our hyperlink evaluations are very successful to improve ranking effectiveness in PageRank-based and HITS-based algorithms.* Table 3 presents the results given by PageRank-based algorithms. Since PROT is least affected by Web Local Aggregation, we consider the collections other than PROT. Compared with the original PageRank algorithm, all new algorithms successfully improve the ranking biased by Web Local Aggregation. The magnitude of the improvement on *ADiff* respect to link evaluations follows the same order: Metric 1 < Metric 2 < Metric 3. Furthermore, the positive values of *HDiff* measure show that the highest ranks of these collections decline. Table 4 presents the results given by HITS-based algorithms. The improvement of new ranking distribution is significant.

**Table 4.** Effectiveness of improved HITS-based algorithms ( $AD(x) = ADiff(x,HT)$ ,  $HD(x) = HDiff(x,HT)$ )

Coll.	Avg(HT)	AD(HTM1)	AD(HTM2)	AD(HTM3)	Top(HT)	HD(HTM1)	HD(HTM2)	HD(HTM3)
MCHIP	7394.9	3140.2	14296.7	16374.6	104	93	-24	279
PROT	40691.4	9603.3	17353.3	26145.8	1	0	607	2478
END98	19288.4	4935.3	8644.1	11957.2	273	291	-12	957
END99	18112.1	4729.9	10611.2	13041.2	211	237	-6	785
END00	18410.0	4776.7	10292.5	14193.1	250	248	-19	868
END01	18064.1	4560.1	12004.8	10513.1	133	167	37	650
ARTG	25825.3	6107.7	4602.6	16812.5	101	89	-4	1826

Our discussion focuses on alleviating the over-ranking problem. As a complement, we present some snapshots of top ranked pages given by improved algorithms as a demonstration how high quality pages win in these ranking systems. Table 5 presents 15 top ranked pages using algorithm PRL2M3 and HTM3. Both of them make good improvement compared with that in Table 1.

**Table 5.** Top ranked pages in PRL2M3 (left) and HTM3 (right)

Rank	URL	Rank	URL
1	www.nyu.edu	1	www.nyu.edu
2	www.nyu.edu/bin/phfnyu	2	www.nyu.edu/bin/phfnyu
3	www.nyu.edu/gsas	3	www.nyu.edu/gsas
4	www.nyu.edu/prospects.nyu	4	www.med.nyu.edu/ethics.html
5	www.nyu.edu/library/bobst	5	www.law.nyu.edu
6	www.law.nyu.edu	6	www.nyu.edu/library/bobst
7	www.nyu.edu/cas	7	www.med.nyu.edu/som/departments.html
8	www.law.nyu.edu/index.html	8	www.nyu.edu/cas
9	www.nyu.edu/presidential.installation	9	www.nyu.edu/athletics/varsity_teams.html
10	www.nyu.edu/athletics/varsity_teams.html	10	www.stern.nyu.edu/acc/about
11	www.nyu.edu/parentsday	11	rmm-java.stern.nyu.edu/jmis/toppage/index.html
12	www.nyu.edu/msep	12	www.med.nyu.edu/webmastermail.html
13	www.nyu.edu/students.nyu	13	www.cims.nyu.edu
14	www.nyu.edu/alumni.nyu	14	www.law.nyu.edu/sitemap
15	www.nyu.edu/parents_guide	15	www.nyu.edu/its

**Acknowledgements**

I would like to thank Professor Ernest Davis for invaluable comments and feedback. He also improved the write-up of this paper a lot. I would also like to thank many anonymous reviewers for providing invaluable suggestions on this work.

**References**

1. R. Albert, H. Jeong, and A. Barabasi. Diameter of the world-wide-web. *Nature*, 401:130–131, 1999.

2. A. Borodin, G. Roberts, J. Rosenthal, and P. Tsaparas. Finding authorities and hubs from link structure on the world wide web. In *Proc. 10th World Wide Web Conference*, 2001.

3. A. Broder, R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins, and J. Wiener. Graph structure in the web. *Computer Networks and ISDN Systems*, 30:309–320, 2000.

4. L. Kleinberg. Authoritative sources in a hyperlinked environment. In *Proc. 9<sup>th</sup> ACM-SIAM Symposium on Discrete Algorithms*, 1998.

5. R. Lempel and S. Moran. The stochastic approach for link-structure analysis (salsa) and the tkc effect. In *Proc. 9th International World Wide Web Conference*, 2000.

6. A. Ng, A. Zheng, and M. Jordan. Stable algorithms for link analysis. In *Proc. ACM SIGIR*, 2001.

7. L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: bringing order to the web. *Stanford Digital Library working paper*, 1997-0072, 1997.

8. Z. Wang. Improved link-based algorithm for ranking web pages. *NYU Computer Science Dept. technical report TR2003-846*, extended version, 2003.

# Effectiveness Evaluation and Comparison of Web Search Engines and Meta-search Engines

Shengli Wu<sup>1</sup> and Jieyu Li<sup>2</sup>

<sup>1</sup> School of Computing and Mathematics, University of Ulster, UK  
s.wu1@ulster.ac.uk

<sup>2</sup> School of Computing and Mathematical Sciences, Glasgow Caledonian University, UK  
jli10@gcal.ac.uk

**Abstract.** In this paper we present the experiments which evaluate the effectiveness of a group of Web search engines and meta-search engines. Experimental results show that on average the performances of selected meta-search engines and search engines are very close, therefore, the claim that meta-search engines are more effective than their counterpart-Web search engines can not be supported. Another observation is that, very often, search engines and meta-search engines are doing much better with short queries than with long queries, which is surprisingly opposite to conventional information retrieval systems. It suggests that Web search services focus on short queries having very few key words probably with special efforts; however, the general search techniques they use for long queries are unsophisticated and further improvement is in demand.

## 1 Introduction

The Internet and the World Wide Web (WWW) is one of the most important and profound creations of humankind. Since its first occurrence in 1993, it has become very popular everywhere in the last decade. Among many Internet-enabled applications and services available today, information retrieval (IR) is very likely to be one of the two primary uses of the Internet (the other is e-mail). Web search services play the most important role for users to identify some previously unknown resources for a particular information need. Web search engines and meta-search engines are two of the three major types of Web search services.

The performances of these search services are concerned by Web users, developers and researchers. There have been quite a few comparative studies on search capability and performance of general Web search engines, for example in [N95a, N95b, L95, CR96, GMS97, L97, LS97, N97, L99, HCTH99, and JSS00]. However, none of them include meta-search engines. Some meta-search engine developers compared their meta-search engines with the Web search engines which their meta-search engine access [G98, GW96]. They claimed that the results in their meta-search engine had a wider coverage and improved precision than those Web search engines involved. As one of the major solutions for Web search service, Web meta-search engine has cer-

tain potential advantages over other solutions. For example, we can expect that the result from a meta-search engine has a wider coverage than any component Web search engine, considering that no Web search engine is able to index all the documents on the Web. But is it true that meta-search engines can achieve better effectiveness than Web search engines? It is an open question.

One of the authors of this paper has been involved in a distributed information retrieval project MIND [MIND], in which merging results from different resources is one of the two major goals. Their research found that the result merging algorithms used by meta-search engines such as ProFusion and MetaCrawler were not sophisticated; further improvement is possible for better performance [WC04]. Therefore, it is an interesting thing to compare the performances of Web search engines and meta-search engines in the “real” Web context.

The rest of this paper is organized as follows: in Section 2 we survey previous study on this topic. We introduce the experimental methodology in Section 3 and present experimental results in Section 4. Section 5 concludes the paper.

## 2 Previous Work

In this section we first survey several major aspects about Web search services, and then focus on the evaluation work done on these search services.

### 2.1 Three Major Types of Web Search Services

Web search services can be divided into three types: Web search engines, Web meta-search engines, and directories.

On the Web, a directory is a subject guide, typically organized by major topics and subtopics. There are general directories, academic directories, commercial directories, portals etc. The best-known directory probably is the one at Yahoo (<http://www.yahoo.com>). Many other sites (e.g., About, LookSmart, and the Open Directory) now use a Yahoo-like directory including major portal sites.

Search engines are used for full-text searches of Web pages. Search engines are best at finding unique keywords, phrases, quotes, and information buried in the full-text of Web pages. Because they index word by word, search engines are also useful in retrieving a large collection of documents. If a user wants a wide range of responses to specific queries, using a search engine is a good way. General search engines are such search engines that cover all areas of knowledge and from an international perspective.

Search research on the Web has a short and concise history. The World Wide Web Worm (WWW) [B94] was one of the first Web search engines. It was subsequently followed by several academic search engines, many of which later became public companies. Google [BP98] is a well-known example. Right now Google, AllTheWeb, MSN, AOL Search, AskJeeves, HotBot, and Lycos are some of the major general Web search engines.

Meta-search engines do not have their own databases. Any query submitted to a meta-search engine is redirected to multiple search engines, then the meta-search engine collects the results from all selected search engines, merges them together, and presents them to the user.

Web meta-search engines began to exist only a little after Web search engines, MetaCrawler [SE96, SE97] and ProFusion [GWG96] were two of the earliest meta-search engines, until now there have been a large number of public Web meta-search engines such as Vivisimo, SavvySearch [DH97], EZ2WWW, Kartoo, SurfWax, Fazzle, MetaFind, MetaEureka etc. In most cases, result merging is a straightforward process, which is done only with ranking (or score) information of all document links from Web search engines. However, for improving the performance, a more complicated and time-consuming method which fetches the original documents and re-ranks them with their own ranking algorithm has been used in several meta-search engines such as Inquirus [LL98] and Mearf [OKK02]. Meng, Yu, and Liu [MYL02] did a survey on meta-search engines in 2002.

To our knowledge, evaluation of Web meta-search engines has not been reported before except for only a few from the meta-search engine developers themselves. To evaluate the performances of meta-search engines and compare their performances with Web search engines' performances are two of the major aims of this study.

In this study, we focus on performance comparison of search engines and meta-search engines, while directories have not been considered.

## 2.2 Evaluation of Web Search Services

Since the beginning, Web search services have always been under development and evolution. The performance of these Web search services is an important issue. Several different methods have been used, which we can classify into the following three types: (1) Survey-based evaluation; (2) Log-based evaluation; (3) Test-based evaluation.

Survey has been the most popular way to evaluate Web search services, conducted by the trade press or computer industry organizations who 'test drive' and then compare search engines on the bases of speed, ease of use, interface design and other features that are readily apparent to users of search engines. For example, [www.searchenginewatch.com](http://www.searchenginewatch.com) conducts surveys periodically on the comparison of online search systems. [www.searchengineshowdown.com](http://www.searchengineshowdown.com), [www.llrx.com](http://www.llrx.com) and others do similar work and publish their search engines comparison results on the Web. Some academic researchers, for example, [N95a, N95b] adopted this method as part of their evaluation methodology.

Another evaluation method is using the logs of the deployed system for the evaluation. The logs record the behavior of a large number of the system's users on queries of their choices. This method is developed on the assumption that the behavior of users on the system can represent the performance of the system. Related study can be found in [JSBS98, JSS00, SWJS00, and ZE97].

In test-based evaluation, different search engines are actually used to retrieve Web pages and their effectiveness in doing so are compared. It resembles the typical IR evaluations that take place in laboratory settings to compare different retrieval algorithms, though Internet shootouts often consider only the first 10 or 20 documents retrieved, whereas traditional IR studies often consider many more.

Query choosing is an important aspect in the evaluation. According to some search engine evaluations carried out previously, 3 kinds of query choosing methods have been reported, which are as follows:

1. Queries were invented by invited expert users in their specialized area, e.g. in [GP99] or designed by experimenters, e.g. in [L95].
2. Queries were drawn from real life, for example, from Web search engine log files [LS97 and CR96].
3. Queries were designed by some organizations. TREC [HCTH99] and NTCIR in Japan [EOIK03] are two examples.

Quite a few test-based evaluation cases have been reported in the literature. In the following let us discuss some of them.

In the United States, U. S. National Institute of Standards and Technology has been holding Text Retrieval Conference (TREC) for over 10 years [TREC]. The aim of these conferences is to evaluate submitted IR systems. Since 2000, a Web track has been introduced.

In Japan, the Japanese National Institute of Informatics has been holding several NTCIR workshops [EOIK03], which are similar to TREC. However, they mainly focus on Japanese IR.

Leighton [L95], Chu and Rosenthal [CR96], and Leighton and Srivastava [LS97] compared the performances of different groups of general Web search engines, while a directory was included in Gordon and Pathak's work [GP99].

There is some evaluation work focus on a particular area or a particular type of network. For example, Wishard [W98] compared several general search engines with queries relevant to earth science, while Stenmark [S99] evaluated the performances of several intranet search engines.

Hawking et al. [HCTH99] presented results for an effectiveness comparison of six TREC systems (systems submitted to TREC) working on the snapshot collection against five well-known Web search engines working over the Web. They found that all five search engines performed considerably worse than the six TREC systems. Their experiments suggested that the standard of document rankings produced by public Web search engines was by no means state-of-the-art.

### 3 Experimental Methodology

In this section we discuss the methodology used in this study: consideration for selecting a group of Web search tools, queries, evaluation criteria and measurement.

Four Web search engines, Google, AllTheWeb, HotBot, and AltaVista, and four meta-search engines, MetaCrawler, ProFusion, MetaFind, and MetaEUREKA were chosen, since most of them were well-known Web search tools.

**Table 1.** Query format used for the experiment

Number (initial number in TREC)	Short Query	Long Query
1(551)	intellectual property	laws regulations protect intellectual property
2(552)	foods cancer patients	foods diets cancer patients
3(554)	home buying	home buying mortgages
4(557)	Clean air clean water	efforts maintain improve clean airclean water
5(558)	U.S. Russian relationship	present relationship U.S. Russia
6(562)	world population growth	outlook world population growth
7(563)	smoking drinking during pregnancy	effects children mothers smoking drinking during pregnancy
8(570)	future occupation	future occupation greatest need statistics projections
9(572)	selecting nursing home	guidelines criteria selecting nursing home
10(585)	tornado basics	basic information tornado
11(589)	mental illness adolescence	studies services offered mental illness adolescence
12(593)	Bush human right China	Bush administrations policy human rights China

In this study, 12 queries, which were a subset of 50 queries in TREC 2002 Web topic distillation task, were used.

An example of the TREC topic (558) is showed as follows:

<top>

<num> Number: 558

<title> U.S./Russian Relationship

<desc>Description:

Describe the present day relationship between the U.S. and Russia.

<narr>Narrative:

Relevant documents details the contemporary status of cooperation and the overall relationship between the U.S. and Russia at present, including business, trade, financial and in-kind assistance, as well as official governmental relations.

</top>

Each topic includes three parts: title, description, and narrative. As in TREC, all parts could be used to form queries, and narrative was used to judge the relevance of the documents to the topic.

For any information requirement (topic), we may construct different queries. Almost every Web search service provides quite a few different options for query submission. Especially some advanced search functions, e.g., phrases, Boolean queries, and proximity search, could be provided. However, simple key words search is supported by almost every search service. According to some surveys [JSB98, JSS00], key words search was the most popular query format used for Web search. Over 90% of the queries submitted to many Web search engines were just one or more words; other kinds of queries were rare. Also as mentioned in Amanda Spink and his colleagues' study [SWJS00], public Web users tended to use few terms, few modified



queries and rarely used advanced search features. Therefore, in this study, we focus on the key words query format.

Two kinds of queries were considered: short queries and long queries. In either case, stop words were removed. Short queries were constructed with all the words in the “title”, while long queries were constructed with all the words in the “title” part and a few selected words from the two other parts. Such a query forming method is based on the following consideration:

1. Since the words in the “title” indicated the most important concepts that were related to the required information, but that was not always the case for the “description”, and especially for the “narrative” part;
2. Using all the words in three parts almost always produce very poor results;
3. Many search engines have a limitation for how many words can be used in a query, for example, Google only allows up to 10 words.

Table 1 shows the words used for all short and long queries.

In this study, Binary judgment (relevant/irrelevant), as in TREC 2002, was used. In addition, we used average precision at several different document levels (5, 10, 15, and 20) for the evaluation. Some other researchers, e.g., [GP99], also used similar measures.

Web page references from sponsors were all neglected in the experiment. A broken link was regarded as irrelevant. For duplicate links, only the first was evaluated, all others were regarded as irrelevant whether they were actually relevant or irrelevant to the query.

All the experiments were done between June 20 and August 20 of 2003.

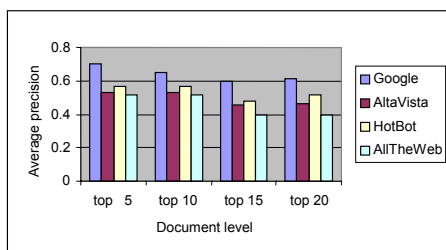
## 4 Experimental Results

The experimental results of four Web search engines are shown in Figures 1 and 2, for short queries and long queries, respectively. On average, Google outperforms HotBot, AltaVista, and AllTheWeb by 21%, 29%, and 42%, respectively for short queries; and it outperforms HotBot, AllTheWeb, and AltaVista by 6%, 34%, and 37%, respectively for long queries.

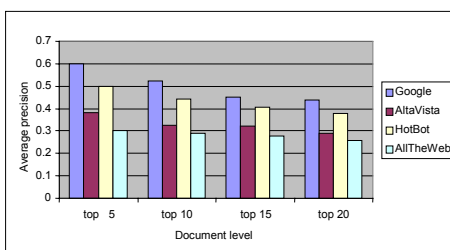
The experimental results of four Web meta-search engines are shown in Figures 3 and 4, for short and long queries, respectively. For both short and long queries, ProFusion is the best and MetaEUREKA is the worst. For short queries, on average ProFusion outperforms MetaFind, MetaCrawler and MetaEUREKA by 5%, 12%, and 21%, respectively. For long queries, on average ProFusion outperforms MetaCrawler, MetaFind, and MetaEUREKA by 7%, 26%, and 35%, respectively.

### 4.1 Comparison between Short and Long Queries

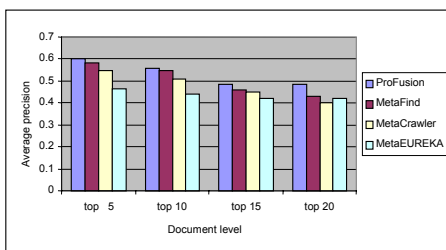
The experimental results are shown for the comparison between short queries and long queries in Figures 5 and 6. For both Web search engines and meta-search engines, short queries lead to much better performance than long queries (37% and 24% improvement for Web search engines and meta-search engines, respectively).



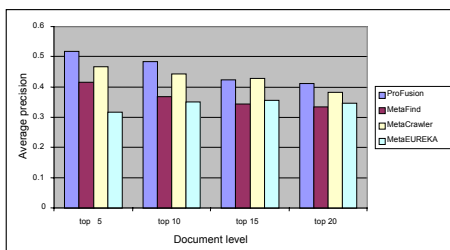
**Fig. 1.** Average precision at different document levels for search engines (short queries)



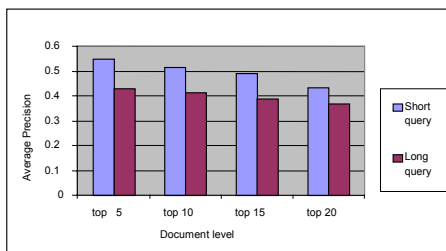
**Fig. 2.** Average precision at different document levels for search engines (long queries)



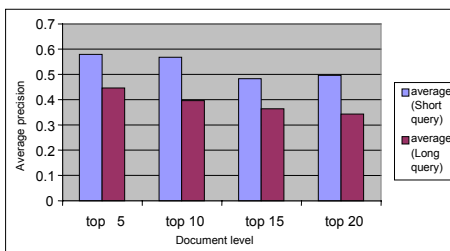
**Fig. 3.** Average precision at different document levels for meta-search engines (short queries)



**Fig. 4.** Average precision at different document levels for meta-search engine (long queries)



**Fig. 5.** Comparison between short queries and long queries for search engines



**Fig. 6.** Comparison between short query and long query for meta-search engines

Besides the above short and long queries, we did some more experiments with more words. Let us discuss this using TREC topic 558 (see Section 3 for its contents). The “narrative” part of the topic includes 34 words, only AltaVista, HotBot, and MetaFind can accept all of them in one query<sup>1</sup>. However, no relevant documents were found in the top 20 links in all these cases. Then we tried 10 words, which were “status cooperation relationship between U.S. Russia business trade financial assistance”. Google returned 5 relevant links, both MetaCrawler and MetaEUREKA returned 2 relevant links, and all the others did not return any relevant links at all.

<sup>1</sup> All other search tools have limitations on the maximum number of words in one query. 10 for Google, and 30 for AllTheWeb, FroFusion, MetaCrawler, and MetaEUREKA.

Compared with the shorter queries “U.S. Russian relationship” and “present U.S. Russian relationship”, in which 8-16 relevant links were found in all 8 search tools, the difference was very big.

Usually, for a given topic, a query with more words leads to a more precise result in conventional IR systems. For example, that is almost always the case in TREC (see also [HCTH99] and [H00]). Why do we observe the opposite phenomena in Web search tools? It suggests that Web search tools concentrate on short queries with very few words, but they may not be as good as some good IR systems for longer, thus more “complex” queries. Probably as indicated in [HCTH99], Web search services do not use start-of-the-art of information retrieval, however, the high effectiveness of these Web search services on short queries is very impressive.

4.2 Comparison between Web Search Engines and Meta-search Engines

In Figures 7 and 8, the performances of the best Web search engine (Google) and best meta-search engine (ProFusion), as well as the average performances of the four Web search engines and four meta-search engines are shown.

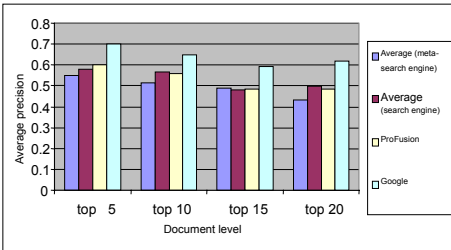


Fig. 7. Performance comparison of Web search engines and meta-search engines (short queries)

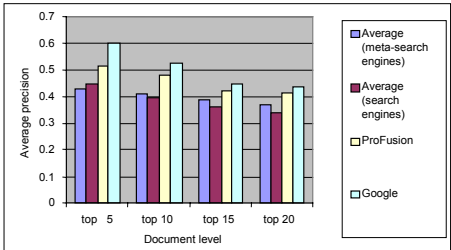


Fig. 8. Performance comparison of Web search engines and meta-search engines (long queries)

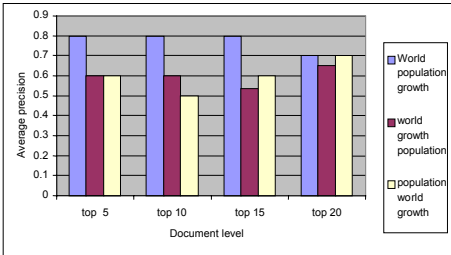


Fig. 9. Search results of Google with different word orders

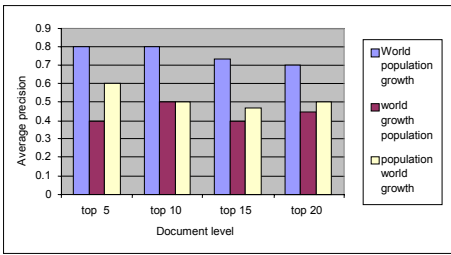


Fig. 10. Comparison between short query and long query for meta-search engines

For short queries, Google outperforms ProFusion by 21%, while the four Web search engines outperform meta-search engines by 7% on average. For long queries, Google outperforms ProFusion by 9%, while the four meta-search engines outperform Web search engines by 4% on average.

The average performances of four Web search engines and four meta-search engines are close; however, none of the meta-search engines is as good as Google, the best Web search engine, in performance. The experiments do not support the claim in [GW96] that meta-search engines are more effective than Web search engines. The performance of a meta-search engine is related to the performances of all component Web search engines and the results merging algorithm. However, surprisingly, ProFusion does not access Google but performs best, while the three others do access Google but are not as good as Profusion.

### 4.3 The Impact of Word Order on Query Results

For key words search, we may organize the same words in different orders. Experiments show that in many cases different orders of words in queries may cause significant differences on query results. Figures 9 and 10 present two examples with MetaEUREKA and Google. “World population growth”, “world growth population”, and “population world growth” were used as three different queries, and the results were quite different. “World population growth” was the one obtaining the best results, since it was exactly the way that appeared in many of the relevant Web pages.

## 5 Discussion

In this paper we have reported the effectiveness evaluation experiment with two different groups of Web search services—Web search engines and meta-search engines.

The experiments have been done with 12 TREC queries and 4 Web search engines (Google, AllTheWeb, AltaVista, and HotBot) and 4 meta-search engines (ProFusion, MetaCrawler, MetaFind, and MetaEUREKA). From these experiments, we have the following observations:

- Though Web search engines and meta-search engines are implemented with different techniques, their average performances are close. The claim that meta-search engines are more effective than Web search engines cannot be supported;
- Google is very likely to be the best among all selected Web search engines and meta-search engines. ProFusion is likely to be the best meta-search engine. However, there is considerable difference between Google and ProFusion;
- Most of the search services are doing very well with short queries including only a few key words, however, when we add more words into the queries, the performances go down, which are opposite to some conventional information retrieval systems. It implies that Web search engines do not use document ranking techniques as well as some conventional IR systems;
- In key words search, the order of these words may affect the performance of the query result significantly. To try to submit a query whose words appear in the same sequence as in the retrieved documents may help improve the quality of the retrieval.

The work in this paper can be furthered in several directions:

- Synonyms (e.g. comparison of searching for “home buy” vs. “property buy”) have not been tested in this study. To find out which Web search tool can support this is an interesting thing.
- In this study, directories have not been included. Though a little more complicated, to compare directories with Web search engines and meta-search engines is an issue that needs attention.
- All Web search engines and meta-search engines selected in this study are general Web search tools. There are a lot of Web search services that are focused on some special topics/fields; to do some evaluation study about them is another direction.
- The query language used in this study is English. Actually, at present many Web search services can support multi-lingual query. To carry out some evaluation with some other languages such as Chinese, Japanese, or Spanish is another direction.

## References

- [AllTheWeb] <http://www.alltheWeb.com>
- [AltaVista] <http://www.altavista.com>
- [B94] O. A. McBryan. GENVL and WWW: tools for taming the Web: hyper search engines. Proceedings of the 1<sup>st</sup> International Conference on the World Wide Web, CERN, Geneva, Switzerland, May 25-27, 1994.
- [BP98] S. Brin and L. Page. The Anatomy of a Large-scale Hypertextual Web Search Engine. *Computer Networks and ISDN Systems*, 30(1-7):107-117, 1998.
- [CR96] H. Chu and M. Rosenthal. Search Engines for the World Wide Web: A Comparative Study and Evaluation Methodology. Proceedings of ASIS 1996 Annual Conference, October 19-24, 1996.
- [DH97] A. Home and D. Dreilinger. Savvysearch: A meta-search Engine that Learns Which Search Engines to Query. *AI Magazine*, 18(2):19-25, 1997.
- [EOIK03] K. Eguchi, K. Oyama, E. Ishida, N. Kando, and K. Kuriyama. Overview of the Web Retrieval Task at the Third NTCIR Workshop. Proceedings of the Third NTCIR Workshop, 2003. Available at <http://research.nii.ac.jp/Web/>
- [GMS97] R. Goldschmidt, C. Mokotodd, and M. Silverman. Evaluation of WWW Search Engines for NIH Implementation. National Institute of Health, Bethesda, MD, USA. January 30, 1997. Available at <http://bigblue.od.nih.gov/Websearch/report.htm>
- [G98] L. Gravano. Querying multiple document collections across the internet. Ph.D. Dissertation. Stanford University, Stanford, CA.
- [GOOGLE] <http://www.google.com>
- [GP99] M. Gordon and P. Pathak. Finding information on the World Wide Web: the retrieval effectiveness of search engines. *Information Processing and Management* 35(2): 141-180, 1999.
- [GW96] S. Gauch and G. Wang. Information Fusion with ProFusion. WebNet 96 Conference, San Francisco, USA, October 16-19, 1996. Available at [www.ittc.ukans.edu/~sgauch/papers/WebNet96.html](http://www.ittc.ukans.edu/~sgauch/papers/WebNet96.html)

- [GWG96] S. Gauch, G. Wang, and M. Gomez. ProFusion: Intelligent Fusion from Multiple, Distributed Search Engines. *Journal of Universal Computer Science*, 2(9):637-648, 1996.
- [H00] D. Hawking. Overview of the TREC-9 Web Track. *Proceedings of TREC 9 Conference*, Gaithersburg, Maryland, USA, November 13-16, 2000. Available at [http://trec.nist.gov/pubs/trec9/t9\\_proceedings.html](http://trec.nist.gov/pubs/trec9/t9_proceedings.html)
- [HCTH99] D. Hawking, N. Craswell, P. Thistlewaite, D. Harman. Results and Challenges in Web Search Evaluation. *Computer Networks*, 31(11-16): 1321-1330, 1999.
- [JSBS98] B. J. Jansen, A. Spink, J. Bateman, and T. Saracevic. Real Life Information Retrieval: A Study of User Queries on the Web. *SIGIR Fourm*, 33(1): 95-97, 1998.
- [JSS00] B. J. Jansen, A. Spink, T. Saracevic. Real Life, Real Users, and Real Needs: a Study and Analysis of User Queries on the Web. *Information Processing and Management*, 36(2): 207-227, 2000.
- [L95] V. Leighton. Performance of four World Wide Web (WWW) Index Services: Infoseek, Lycos, WebCrawler, and WWWorm. Available at <http://www.winona.msus.edu/services-f/library-f/Webind.htm>
- [L97] A. Lange. Sorting through Search Engines. *Web Techniques Magazine*, 2(2), 1997.
- [L99] M. Ljosland. Evaluation of Web search engines and the search for better ranking algorithms. *SIGIR 99 workshop on Evaluation of Web retrieval*, Berkeley, USA, August 19, 1999.
- [LL98] S. Lawrence and C. Lee. Inquiries, the Neci Meta Search Engine. *Proceedings of the 7<sup>th</sup> International World Wide Web Conference*, 95-105, Brisbane, Australia, April 14-18, 1998.
- [LS97] H. Leighton and J. Srivastava. Precision among World Wide Web Search Engines: Alta vista, Excite, Hotbot, Inforseek, and Lycos. June, 1997. Available at <http://www.winnoa.edu/library/Webind2/Webin2.htm>
- [MetaCrawler] <http://www.meracrawler.com/>
- [MIND] The MIND project: <http://www.mind-project.org/>
- [MYL02] W. Meng, C. YU, and K. Liu. Building Efficient and Effective Meta-search Engines. *ACM Computing Survey*, 34(1): 48-89, March 2002.
- [N95a] G. Notess. Searching the World Wide Web: Lycos, WebCrawler and More. *Online*, 19(4), 48-53.
- [N95b] G. Notess. The InfoSeek Database. *Database*, 85-87.
- [N97] B. Nance. Internal Search Engines Get You Where You Want to Go. *Network Computing Online*, October 8, 1997. Available at <http://www.nwc.com/819/819cn2.html>
- [OKK02] B. Oztekin, G. Karpis, and V. Kumar. Expert Agreement and Content Based Reranking in a Meta-search Environment Using Mearf. *Proceedings of the 11<sup>th</sup> International World Wide Web Conference*, 333-344, Honolulu, Hawaii, USA, May 7-11, 2002.
- [ProFusion] <http://www.ProFusion.com/>
- [S99] D. Stenmark. A Methodology for Intranet Search Engine Evaluation. Available at <http://w3.informatik.gu.se/~dixi/publ/method.pdf>
- [SE96] E. Selberg and O. Etzioni. Multi-service Search and Comparison Using the MetaCrawler. *Proceedings of the 4<sup>th</sup> international WWW conference*, Paris, France, May 6-10, 1996.
- [SE97] E. Selberg and O. Etzioni. The MetaCrawler Architecture for Resource Aggregation on the Web. *IEEE Expert*, 12(1):8-14, 1997.
- [TREC] <http://trec.nist.gov/>
- [W98] L. Wishard. Precision Among Internet Search Engines: An Earth Science Case Study. *Spring* 1998. Available at <http://www.library/ucsb.edu/istl/98-spring/article5.html>

- [SWJS00] A. Spink, D. Wolfram, B. J. Jansen, and T. Saracevic. Searching the Web: the Public and Their Queries. Available at <http://jimjansen.tripod.com/academic/pubs/jasist2001/jasist2001.html>
- [WC04] S. Wu, F. Crestani and F. Gibb. New Methods of Results Merging for Distributed Information Retrieval. Distributed Multimedia Information Retrieval (Lecture Notes in Computer Science 2924, Springer) 84-100, 2004.
- [ZE97] O. Zamir, O. Etzioni, O. Madani and R. Karp. Fast and Intuitive Clustering of Web Documents. Proceeding of the ACM SIGMOD International Workshop on Data Mining and Knowledge Discovery, 287-290, Montreal, Canada, June, 1997.

# Semantic Search in the WWW Supported by a Cognitive Model

Katia Wechsler<sup>1</sup>, Jorge Baier<sup>1</sup>, Miguel Nussbaum<sup>1</sup>, and Ricardo Baeza-Yates<sup>2</sup>

<sup>1</sup> Departamento de Ciencia de la Computacion,  
Pontificia Universidad Católica de Chile,  
Casilla 306, Santiago, Chile  
kwechsle@puc.cl, {jabaier,mn}@ing.puc.cl

<sup>2</sup> Departamento de Ciencias de la Computacion,  
Universidad de Chile,  
Blanco Encalada 2120, Santiago, Chile  
rbaeza@dcc.uchile.cl

**Abstract.** Most users of the WWW want their searches to be effective. Currently, there exists a wide variety of efficient syntactic tools that have can be used for search in the WWW. With the continuous increase in the amount of information, effective search will not be possible in the future only with syntactic tools. On the other hand, people have remarkable abilities at the moment of retrieving and acquiring information. For example, a librarian is capable of knowing, with great precision, what a client seeks by asking a small set of questions. Motivated by the efficiency of that process, we have created a web search system prototype based on ontologies that uses a cognitive model of the process of human information acquisition. We have built a prototype of a search system whose output better meets the expectations of the users compared to tools based only on syntax. Using this model, the prototype “understands” better what the user is looking for.

**Keywords:** Semantic Search in the WWW, Cognitive Models, Semantic Web.

## 1 Introduction

The explosion in the amount of information in the WWW occurred in the last years is one of the central problems faced by its users. The amount of information is so big that it is necessary to possess efficient and effective mechanisms of information recovery.

Currently, one of the most commonly used tool for search is information filtering [1], which consists of filtering the information returned by the search process based on certain parameters. Among these parameters are the user’s profile [2], which is a definition of characteristics that represent the interests of the user that enables to delimit his search, such as likes, language, and interests. Another parameter frequently used is ranking, that is a score given to the web pages and depends on how many times the page has been chosen by users of the search engine. Another tool used for search is indexation [3,4], that consists of elimination of frequently used terms and search of roots of words or synonymous, to create an index that represents a document.



All the methods described above use syntactic handling of words, in the sense that all information used from words depends exclusively on the word itself. Nevertheless, this is not sufficient; most users get frustrated when the result of their search includes plenty of pages that have nothing to do with their interests. Moreover, the amount of available information increases and systems based on syntactic methods will soon be surpassed. It is necessary, then, to add semantic elements to search systems. In this work we are concerned on how to incorporate the sense that users give to words into a search algorithm.

The absence of semantic capabilities imply that search engines return irrelevant information with words with the same syntax, but that do not mean the same thing, words that mean the same thing but appear in a context different to the user's, and leaving out those that mean the same thing but are spelled differently and appear in the same context.

An example of a semantic search system is OBSERVER [5], which uses specific ontologies that enable users to express the demanded information at an abstraction level beyond words itself. To this end, it keeps a set of small ontologies, each of them associated to a set of documents. When a requirement is demanded, the system chooses the ontology class which meets the user's requirements and asks the user for confirmation, then it returns the documents associated to that class.

On the other hand, it is remarkable the ability of the human being when look for information. For example, a librarian is capable of knowing, with great precision, what a client looks for by asking a very small set of questions. This is due, among other things, to the fact that humans easily understand the language, context and motivation of the person who asks for his help. The cognitive process of information has been studied by diverse authors [6], nevertheless, according to our knowledge, there have not been attempts of integrating cognitive models into automated search systems.

In the following sections, we describe a search prototype based on ontologies that uses a cognitive model of the human process of information acquisition. Section 2 presents a description of tools used in the prototype development, section 3 describes a cognitive model for human understanding, section 4 explains the prototype and shows a real example using ontologies of public domain and, finally, in section 5, we comment on some preliminary results and sketch our future work.

## **2 Ontologies, Web Documents, and Information Extraction**

The following subsections describe ontologies and how they are related to web documents. Moreover, we describe the process of information extraction, which is central to our algorithms.

### **2.1 Ontologies and Web Documents**

Ontologies provide a way to represent and share knowledge using a common vocabulary; they define a protocol of communication and allow knowledge reuse. Ontologies are useful to represent both concepts and the relationships among them.

The elements that compose an ontology are *classes*, which are a formalization of concepts; *relations* that represent interactions among classes; *functions*; *Instances*, that are objects of a class; and *axioms*, that are declared logical truths that hold about elements of an ontology.

A principle of well-designed ontologies is that similar concepts be grouped together or represented using same primitive [7]. This principle is central to this work, since the closer the concepts in the graph, the more related they are.

Ontologies are not related to the web in a natural way; therefore, additional efforts are necessary to relate them. There exist two ways of relating them: the first one, is that every web page contains a small ontology that represents its content (this ontology can be written in any of RDF, OIL or DAML+OIL languages [8]). This approach gives too much freedom to the user to describe his pages, therefore making it difficult for a computer system to analyze its content. Furthermore, common users of web pages are generally unenthusiastic to use tools that define semantically well their pages.

The second way of relating web pages to ontologies, is to have a unique ontology in which every class is linked to a database of web information (web pages, pdf, images, etc.). This is achieved by creating databases containing tuples of documents associated with the classes of the ontology. In this work, we use this approach.

Currently, the CYC project [9], is a research project carrying out construction of ontologies. The CYC project is still in development, and its goal is to construct a knowledgebase containing a significant part of commonsense knowledge of this century.

## 2.2 Information Extraction of a Document in Natural Language

Whilst information recovery is used to retrieve documents that contain particular words, an information extraction (IE) algorithm allows a user to obtain all the concepts that could be associated to the words in the text [10].

There exist several tools that allow IE; among the GATE software [11]. These systems can work also with compound terms. Thus, such a system would be able to recognize the compound term "President of Chile" as one concept. In case a word or compound term is associated to more than one concept, all the associated concepts are returned.

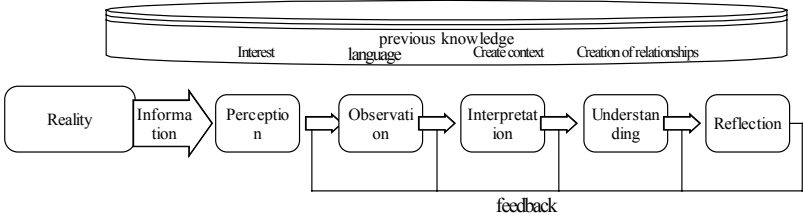
In our prototype we use a Java library for IE known as Stand-Alone Gazetteer [12]. We have adapted Stand-Alone Gazetteer to work with the CYC ontology.

## 3 A Cognitive Model for Human Understanding

The cognitive model of information extraction (fig. 1) [13,14,15,16] consists of 5 parts: perception, observation, interpretation, understanding and reflection.

The first part of the process addresses the way information is acquired from reality, i.e. environment of the human being. We define *perception* as the act of capturing the information in the reality. Human beings have a finite capacity of storing and gathering perceived information. Humans only perceive information of their interest. That

information is determined by what is relevant for the person. For example, someone who has interest in plants, after entering to an office, would perceive the existence of them. On the other hand, one that does not have any interest in plants, would not notice them.



**Fig. 1.** Stages of the model

Perception is the act of describing an object with a language. Not all the things that are perceived are observed. For the observation of an object to occur there is a need for a language that allows to describe it. Two persons who dominate different languages will observe different things while looking at the same object. In this way, while looking to a plant, a gardener sees more things than a person who does not dominate a language that describes the plants in detail.

The interpretation occurs when the observed object is contextualized by the observer. Using the description of the object, the observer places it in a context and is able to relate it to other elements that belong to the same context. For example, if the gardener sees a plant he may know what type of plant is, if it is eatable, ornamental, etc. On the other hand, if an interior decorator sees the plant, he will be able to know where to place it, for what types of decorations it is suitable to, etc.

Understanding arises once information has been interpreted. It is a process that is strongly linked to what the observer *can do* with his interpretation in order to validate or invalidate his observation. For example, once a plant is observed, the gardener will know if it lacks light or water and will do what is necessary. The act of understanding is related with doing. Often, when a person understands something, it acts according to this understanding.

Reflection is the last part of the model. It consists of checking the whole process and determining if the actions of the observer were adequate. For example, the gardener will be able to see if the faded plant has recovered after watering it. Otherwise, he understands that it was not lacking water but another element. Thus, he increases the possibilities of succeeding in the future by repairing the information of the mistaken component.

## 4 Search System Prototype Based on the Cognitive Model

This section presents a search system prototype based on the cognitive model presented in the section 3. The prototype is based on the ontology *cyc.daml* of the *CYC* project.

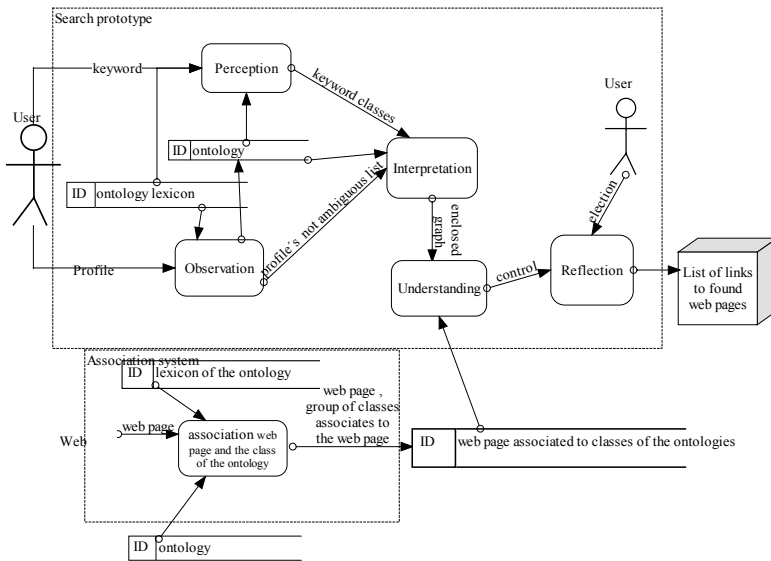


Fig. 2. Design of the developed prototype

Figure 2 shows the search system design based on the model of section 3. It is composed by five modules that interact with the user, the ontology, or the ontology lexicon, and give as result the feedback of the system and the web pages related to the search.

Besides from the system of search, there is another subsystem: the association system. Its goal is to associate the web pages with the classes of the ontology, keeping a database of pages associated to ontology classes. It must be working all the time, constantly updating its associations.

The following subsections explain and exemplify the algorithms of each of the modules.

#### 4.1 Step 1: Perception Level

The aim of this step (first step of figure 2) is to determine what is the motivation of the system, i.e., what the system “wants” to look for. The algorithm is shown in figure 3.

```

Asks the user for a list of words
Keyword ← classes of the ontology associated those words
    
```

Fig. 3. Algorithm for perception level

For example, let us suppose that the system receives the word “shape”. This word is associated with four classes of the CYC ontology: GeometricallyDescribableThing, ShapeType, ShapingSomething, shapeOfObject, which are stored in the array *Keyword*. This array is the output of the module.

## 4.2 Step 2: Observation Level

The observation level reproduces the human ability to describe a perception by means of language. In our system, it is necessary to know the language<sup>1</sup> of the user to understand what information is he really interested in. The system knows the language of the user by asking him to describe his interests using a paragraph in natural language.. Then, the system turns the paragraph into a list of references to classes of the ontology that represents the interests of the user.

```

List ← {<class,word> | word is in Paragraph given by user and class
           is returned by Gazetteer[word]}
Profile_repeated ← <class, word> elements of List such that word has
more than one associated class in List.
Profile ← elements <class, word> of List such that word has a unique
associated class in List.
for each element <class, word> in Profile_repeated
    Evaluation[class, word] ← classes(nodes) of the smallest
subgraph of the ontology generated using a BFS search
starting from class and that contains an element in
Profile.
endfor
for each element <class, word> in Profile_repeated
    relation_index[class, word] <- relation index of class with re-
gard to Evaluation[class, word]
endfor
for each word wd such that there exists <class,wd> in Pro-
file_repeated
    Add to Profile Argminclass relation_index[class, wd]
endfor

```

**Fig. 4.** Algorithm for observation level

When the Gazetteer recognizes a word, there exist 2 possibilities: that the word is associated with exactly one class of the ontology or that it is associated with more than one class. In spite of having more than one ontological meaning, the latter type of words have only one meaning for the user, which should be determined by the content of the paragraph (the context given by the paragraph). Due to this, for each of these words, the algorithm must determine a unique class of the ontology associated to it, considering the content of the whole paragraph.

To this end, the algorithm separates the words associated with only one class of the ontology and it places these classes in the vector *Profile*. Words associated with more than one class, are placed in *Profile\_repeated*, paired to every class to which they are related.

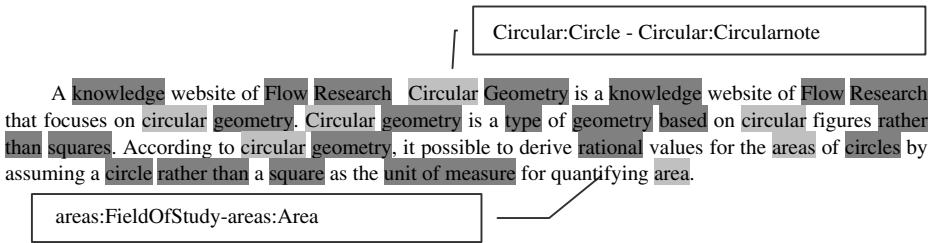
To determine the unique class associated with every word in *Profile\_repeated*, we choose the class that is semantically closer and more related to the elements of *Profile*. To measure how close and related is a particular class with the elements of *Profile*, the algorithm uses an *relation index*. To calculate this index, it takes into account two values. The first one, is the minimum distance between the class and some element of *Profile* (this distance can be calculated directly from the information in the

---

<sup>1</sup> The word language here must be understood in the same way as in section 3, i.e. as the means the person has to describe his knowledge.

matrix *Evaluation*). The second one, is the number of classes of *Profile* that are related to the class at minimum distance. Finally, the relation index is computed as the quotient between these two quantities.

For every word that is associated with more than one class of the ontology, the algorithm chooses the class with the minimum relation index and adds it to *Profile*. In this way, we obtain a list of non ambiguous classes in *Profile* for a given paragraph.

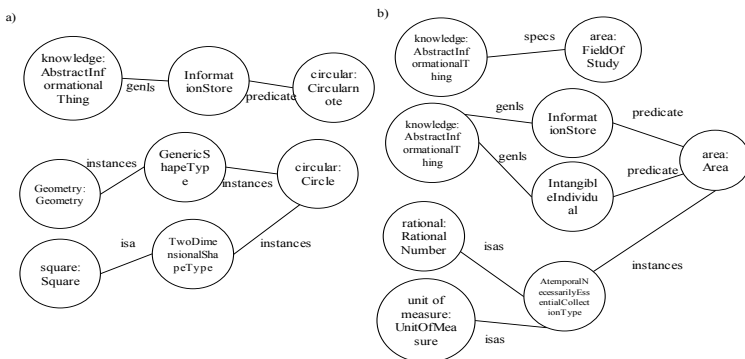


**Fig. 5.** Paragraph of interests that shows the words that are present in the lexicon

For example, suppose the user enters the paragraph shown in figure 5. Words in light gray have more than one associated class in the ontology CYC, whereas the words in dark gray only have associated class (words not marked are not known by the ontology).

Figure 6 shows how related are the concepts associated to the words *circular* and *area* with the rest of the words in the paragraph. In the graph, it is clear that *area* and *circle* are the classes that really correspond to the meaning given in the paragraph, since the relation index is the minimum (the relation index of circularnote is 1, whereas that of circle is  $\frac{1}{2}$ ). Thus, we leave in the profile the concepts: Circle, Area, discarding FieldOfStudy and Circularnote.

For this algorithm to work correctly, it is essential that there exist words associated with only one class of the ontology in the paragraph provided. Although this is an important limitation, it is difficult to find paragraphs that describe areas of interest and that are completely ambiguous. In case that this situation arises, the algorithm outputs an empty profile



**Fig. 6.** (a) Search of the class most related to the word circular (b) Search of the class most related to the word area

### 4.3 Associating Web Pages to Classes of Ontology

For the search system to work, it is necessary to have a database of web documents associated with concepts in the ontology (shown in figure 2). To construct the database, the module uses the algorithm described in the step above. Thus, it receives a document as input (which is obtained by means of a web spider) and one association between the webpage and each class found in the document is stored in the database.

### 4.4 Step 3: Interpretation

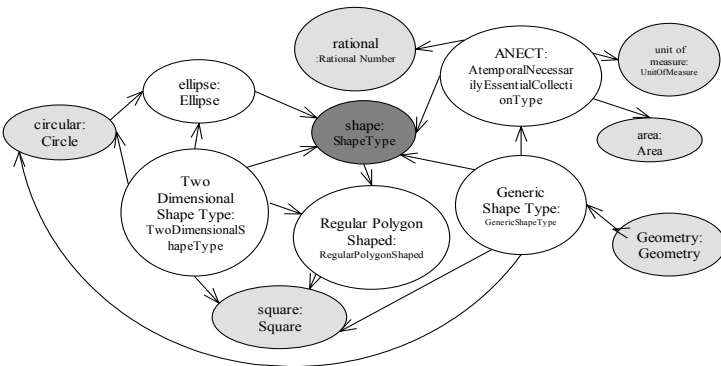
In this step (interpretation module in figure 2) the keyword is contextualized, i.e., the algorithm finds the context of the word with respect to the user. Specifically, the algorithm finds the ontology subgraph that is semantically closer to both the keyword and the elements in the profile. This resulting subgraph corresponds to the *area of interest* of the user or to the context that the user give to the words.

The area of interest is computed using a breadth-first search, starting from all nodes associated to the keyword until an element of the *Profile* is found or until search has reached a *limit* depth<sup>2</sup> (which in practice is 5).

### 4.5 Step 4: Understanding

This step (understanding module in figure 2) shows to the user in a useful manner the set of shortest paths from the keyword to the profile classes. Since it is possible that the user wants to search out of the area of interest, the algorithm also shows the nodes directly connected to the keyword .

In case the resulting graph is too complex to be visualized, the graph is pruned using a transitivity rule. For example, if *a* is subclass of *b* and *b* is a subclass of *c*, the intermediate relation (*b*) is hidden and a direct arc between *a* and *c* is shown.



**Fig. 7.** Resulting Graph showed to the user. The dark gray node corresponds to the keyword and the light gray ones to the closest related elements in the profile.

<sup>2</sup> We have observed that at a greater distance, classes are generally in a context that has no relation with the search.

Now the system lets the user interact with the graph. The user can click over the nodes in the graph. Afterwards, the system shows all the web pages associated to the classes of the ontology clicked by the user. In our example, in case the user clicks the classes Shapetype and ellipse, the system will show links to the web pages that contain both concepts.

#### 4.6 Step 5: Reflection

At this point (module reflection of figure 2), the system needs feedback from the user to verify that it has done the correct thing. If the user chooses one or more of the shown classes, it means that he agrees with what was displayed by the algorithm, that is to say, the concept he was looking for has been interpreted well by the system.

If the user chooses one of the nodes that do not belong to the paths between keyword and profile nodes, it means he does not agree with the result of the system. This can be because the concept was not interpreted well or the context found by the algorithm was not the correct one. At this point, an option is presented to the user in order to allow him to better customize his profile. The system offers him to add new concepts by changing the paragraph that defines his language.

### 5 Some Preliminary Tests

In order to test our system we compared it against the well-known search engine Google. The experiment consisted in searching for information about the Isle of Man (an country located in the Irish Sea).

We invoked Google with the keyword “Man”, and kept the first 44 results. From these, only 6 were related to the island (13.6% of efficacy). We processed these 44 pages with our system, generating a database that contained links to 742 different classes of the ontology. Afterwards, we invoked our system with the keyword “Man” (which in fact has three associated concepts in the ontology: “AdultMalePerson”, “ControllingSomething”, and “Country”). Furthermore, we entered a profile with two classes: “Nation” and “GeopoliticalEntity”. In the resulting graph, when the user clicks over the node “man” he obtains a list of 18 links. Among them, 6 are related to the isle of Man (33% of efficacy).

In this simple test, we see that the efficacy of our prototype is better than that of Google. Moreover, since our prototype “discovered” that what the user wanted was information of the isle of man, the information returned is fewer (only 18 from a total of 44 possible answers).

### 6 Conclusions and Future Work

The system presented is not a purely semantic web search engine, nor a syntactic one. The main advantage of our approach is that it can work with existent web technology, since it is not necessary to add any special context to HTML pages. Nevertheless, its main limitation is that it relies on the existence of a big ontology, containing, ideally,



all human commonsense knowledge. Although the existence of such an ontology could be regarded as an utopia, we think it is more promising that pure semantic web, since it is unlikely that common web authors will be able to (or want to) use advanced tools to semantically describe their pages.

One of the main contributions of this work corresponds to the use of a cognitive model of the way a human being retrieves information. This model has been enlightening to us, especially with regard to the design of the system.

Currently, we are doing extensive tests of efficacy and efficiency of the system. Furthermore, we plan to improve the efficiency of the observation step, replacing breath-first search by a fast algorithm that will use a database of pre-computed distances between ontology nodes.

## References

1. Joaquin Delgado, Naohiro Ishii: Multi-Agent Learning in Recommender Systems for Information Filtering on the Internet. *Int. J. Cooperative Inf. Syst.* 10(1-2). (2001) 81-100
2. Gediminas Adomavicius, Alexander Tuzhilin: Using Data Mining Methods To Build Customer Profiles. *Computer Innovative Technology For Computer Professionals*, (2001) 74-82
3. William B. Frakes, Ricardo Baeza-Yates: *Information Retrieval*. Prentice Hall. (1992)
4. Ricardo Baeza-Yates, Berthier Ribeiro-Neto: *Modern Information Retrieval*, Addison Wesley. (1999)
5. Eduardo Mena, Vipul Kashyap, Amit P. Sheth, Arantza Illarramendi: OBSERVER: An Approach for Query Processing in Global Information Systems based on Interoperation across Pre-existing Ontologies. *CoopIS 1996*. (1996) 14-25
6. Perkins, D.N.: What Is Understanding? In M. S. Wiske (Ed.), *Teaching For Understanding: Linking Research With Practice*. San Francisco. (1998) 39-57
7. Van Heijst, G., Schereiber, A.T. Y Wielinga, B.J.: Using Explicit Ontologies In Kbs Development. *International Journal of Human and Computer Studies* (1996) 183-292
8. The DARPA Agent Markup Language Homepage, <http://www.daml.org>
9. Overview of OpenCyc, <http://www.cyc.com/cyc/opencyc/overview>
10. Appelt, D.: An Introduction to Information Extraction. *Artificial Intelligence Communications*, 12(3). (1999) 161-172
11. Atanas Kiryakov, Borislav Popov, Damyan Ognyanoff, Dimitar Manov, Angel Kirilov, Miroslav Goranov: Semantic Annotation, Indexing, and Retrieval. *International Semantic Web Conference (ISWC)*. (2003) 484-499
12. Ontotext – a Sirma Lab for Knowledge and Language Engineering, <http://www.ontotext.com/>
13. Gardner, H.: Multiple Intelligences Approaches To Understanding. In *The Disciplined Mind: What All Students Should Understand*. Simon & Schuster, New York. (1999) 186-213
14. Gardner, H.: Perspectives Of Mind And Brain. In *The Disciplined Mind: What All Students Should Understand*. Simon & Schuster, New York. (1999) 60-85
15. Guilford, J.: *The Nature Of Human Intelligence*. McGraw-Hill, New York. (1967)
16. Perkins, D. N.: What Is Understanding? In M. S. Wiske (Ed.), *Teaching For Understanding: Linking Research With Practice*. San Francisco. (1998) (39-57)

# Semantic Board: A Bulletin Board System Supporting Dynamic Semantic Information

Eui-Hyun Jung

Dept. of Digital Media, Anyang University,  
708-113, Anyang 5-dong, Manan-Gu, Anyang City, Kyunggi-do, Korea  
jung@anyang.ac.kr

**Abstract.** Although a bulletin board system has been an important virtual place for exchanging information in the same interest group for a long time, its users have to browse each article for getting target information because current board system has not been able to provide conceptual data access. To solve this issue, we propose a semantic-based bulletin board system, called Semantic Board that supports tagging and searching articles using semantic information. Without modification, it can be adopted in various application fields using different semantics with a proposed ontology named SBoard. The SBoard ontology is designed to describe RDF predicates representing semantic information in the Semantic Board. The ontology indicates the form generation in the frontend and RDF data access in the backend of the target predicates. Using this, the Semantic Board is able to generate dynamically HTML forms to ease difficulties in writing semantic tags manually and store users' input as a corresponding RDF document. The SBoard is also used to generate RDQL statement for searching articles based on semantic information.

## 1 Introduction

A bulletin board system has been a killer application dedicated to sharing information for online community before the Web arrived. A lot of network services such as the Gopher have been dismissed after the emergence of the Web, but the board system survived and has been integrated into the part of the Web, so called a Web-based board system. Since the board system enables users to share information and exchange their opinion easily, a number of studies have been conducted on various fields such as groupware [1], official document processing [2], and education system [3]. For other perspective view, some researchers concentrated on extending its functionalities including multimedia support [4], or collaboration filtering [5].

In spite of its convenience and effectiveness, existing board systems tend to take a lot of time and efforts from users for getting target information. Though some advanced board systems provide searching function based on text matching, its data access is processed at the lexical level. The lexical level access to the board system reveals a limit on getting proper information because same meaning can be represented in different ways in the board systems. For example, a date of 6<sup>th</sup>.Jul.2003 can

be represented as either 2003.7.6 or 03-07-05. If these expressions were contained in a "job hunting board," job hunters would suffer to search a proper article including correct date that they want to get.

This issue is not confined only to the board system, but also to the other information systems because current information systems including the Web have used lexical-based information processing other than semantic-based. Search engines on the Web provide users with convenient searching functions, but their search results are mainly extracted by using the keyword matching. Authors on the Web create web pages with their whims and use subtly different terms that can fool a simple keyword search [6]. For this reason, a lot of researchers are interested in the Semantic Web as a solution to this issue [7][8]. Within the Semantic Web, information can have semantic contents that can be processed by machines or agents [9][10], so the Semantic Web will enable intelligent service [10] and provide more accurate search result satisfying users' intention.

The main methodology of making the Semantic Web is a semantic tagging to the data on the Web. The semantic tagging means activity of which authors tag the semantic information manually to their resource using the Semantic Web tag language like Resource Description Framework (RDF) [11] and related ontologies. For this purpose, W3C and other researchers suggested several Semantic Web tag languages [12] and applications such as Protégé [13] and ADF[14] are announced.

The manual semantic tagging is suitable to the Web pages, but this approach is not preferable to the board system because it requires a great amount of knowledge about the Semantic Web tag languages and it may cause frequent typos. An easy alternative is giving HTML form to users to collect user's semantic data input, but this method causes a severe scalability problem. The input data on the HTML form has to be processed by the Common Gateway Interface (CGI) and general CGI code can only processes predefined data. It makes the board system can only support fixed semantic data. If this approach is adopted, a new board system has to be developed whenever different semantic data is required. Therefore a desirable semantic-based board system not only provides easy semantic data access from the Web browser, but also supports various ontologies without modifying its processing code.

To satisfy these requirements, we propose a new board system, called Semantic Board that can dynamically generate posting/query forms for the target semantic information without code modification. It can also provide a storage access module for creating a RDF document from the input data and a search module generating RDF Data Query Language (RDQL) [15] statement. With the proposed ontology named SBoard, the Semantic Board can be used in various application fields. The SBoard ontology is designed to write a Predicate Semantic Description (PSD) that describes RDF predicates imported in the Semantic Board.

The paper is organized as follows. Section 2 will discuss building blocks and challenges for creating a semantic-based board system. Section 3 addresses the architecture of the SBoard ontology and the PSD. Section 4 focuses on the internal structure of the Semantic Board. Section 5 concludes this paper.

## 2 Building Blocks and Challenges

### 2.1 The RDF in Describing Semantic Information

Although the several Semantic Web description languages have been announced to describe semantic information on the Web, the RDF can be a suitable solution for a semantic-based board because of its simple yet powerful description ability. The RDF is based on the idea of describing resources in terms of simple predicates and predicate values [11]. This relationship could be represented as the RDF graph shown in Fig.1.(a).

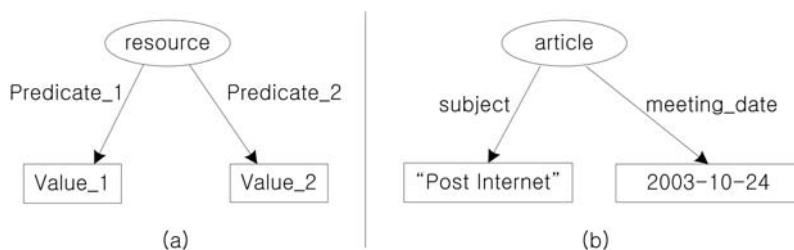


Fig. 1. Examples of RDF graph

In a semantic-based board, a targeted resource will be an article that a participant posted. Therefore the article's URI, predicates and predicate values should be provided to create a RDF/XML document that holds an article's semantic information. For example, to announce conference meetings, the article in the board may have predicates such as a subject, meeting date shown in Fig.1.(b).

### 2.2 HTML Forms and the RDF

A semantic-based board should provide an HTML form to keep its users from writing RDF/XML manually. When using this approach, the data on an HTML form needs to be mapped to the elements of a RDF document to get the article's predicates and their values. When the data on an HTML form is transferred to the Web server, form data is encoded as a stream of "name=value" pairs separated by the "&" character and it is processed by corresponding CGI code. Fortunately, each "name=value" pair can be easily mapped to "predicate=value" of semantic information. That is, the input data on an HTML form can be simply processed as elements of RDF document with the CGI code like other Web applications. This seems an intuitive solution for making a simple semantic-based board.

### 2.3 Issues and Requirements

Above CGI based approach reveals a severe scalability problem for a semantic-based board. In the CGI programming, an HTML form and its processing code are hard-coded in the board system before the board is released. That means the attached se-

mantic information has to be fixed when the CGI programming is simply applied to an semantic-based board. If a board manager wants to change the board's semantic data schema, the board system's HTML form and its processing code should be modified or rewritten because the frontend and backend of the Web application is adhered to the predefined data.

In the RDF, the different semantic domain requires different predicates. For example, a domain related to photos will require the predicates such as resolution and picture type, but other domain related to job hunting may need the predicates including job category and opening date. By this reason, a semantic-based board supporting a single application domain can not be reused for other domains unlike ordinary board systems.

To support various predicates without code modification, a semantic-based board system should satisfy two requirements at least. First, it should have a method to describe target predicates. Especially, the method should indicate the elements of corresponding HTML form and RDF document for each predicate. Secondly, the board system is able to read the described predicates and generates frontend HTML forms and backend processing codes. To satisfy these requirements, we suggest a Semantic Board and new ontology.

### 3 SBoard Ontology and Predicate Description

#### 3.1 Elements of Ontology Design

We designed a new ontology named SBoard to be used for describing RDF predicates that explain articles' semantic data. Ordinary ontologies are designed to describe resources on the Web, but the purpose of the SBoard ontology is to describe the predicates of other ontologies that will be imported into the proposed semantic-based board. Therefore a predicate of the other ontologies is a resource in the SBoard. The vocabulary of the SBoard is designed to indicate HTML form and RDF document template for the predicates of other ontologies.

The predicates of the SBoard are categorized into three groups; Posting-Form group, Query-Form group and RDF data group. The predicates in the Posting-Form group are *formtag*, *label*, *range* and *default*. These predicates are used to construct corresponding HTML form component in the posting page. The Query-Form group's predicates are *label* and *target*, which make HTML form components in the query page. The predicates of RDF data group are *datatype* and *prefix* that provide information needed to create RDF document. The detailed description of SBoard predicates is shown in Table 1.

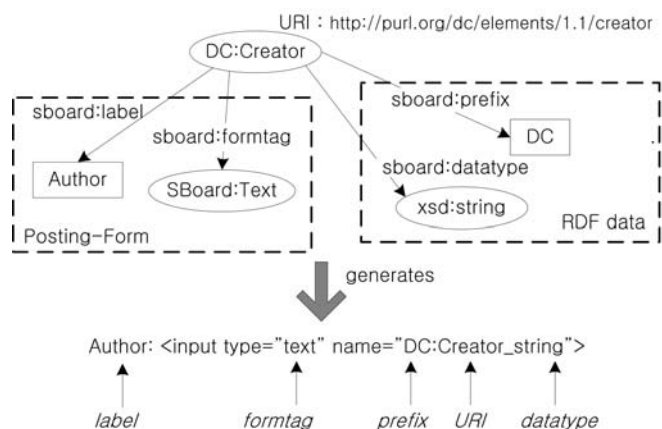
#### 3.2 Predicate Semantic Description

The Semantic Board is able to import a description file containing predicates. This predicate description file will be called as a Predicate Semantic Description (PSD) in

**Table 1.** Predicates of the SBoard Ontology

Group	Name	Mandatory	Description
Posting-Form	formtag	yes	indicates the HTML form component such as TEXT
	label	yes	indicates the LABEL before the component
	range	optional	indicates the data when formtag is SELECT component, RDF:Bag is used to set the data
	default	optional	indicates the default value on the component
RDF data	datatype	yes	indicates the data type the predicate can have
	prefix	yes	indicates the namespace prefix for RDF document
Query-Form	label		indicates the LABEL before the component
	target		indicates the target predicate for searching

this paper. A PSD is consisted of Common part and several Predicate parts. The Common part has a *modelname* predicate for connecting backend database access and it exists only once in a PSD. The Predicate part is applied to each predicate. In the Predicate part, RDF data group's predicates provide information for creating RDF document. Posting-Form group's predicates indicate information to generate HTML form component for posting an article. The predicates of Query-Form group are similar to the member of Posting-Form group, but these generate HTML form for query input.

**Fig. 2.** Predicate part of DC:Creator and its expected results

Each Predicate part in a PSD is used to generate either a posting form or a query form. To generate a posting form, the predicates of Posting-Form and RDF data group are required. For example, if a board manager wants to use “Creator” predicate from the Dublin Core [16] for his board system, the description and its results may be as shown in Fig.2.

In the case of DC:Creator, it needs a text input component and it should be stored as xsd:string data type in a RDF document. A *sboard:label* and *sboard:formtag* in the

Posting-form group indicate HTML form component. The *sboard:prefix* is required to write this predicate for the name space in the RDF document. If input data is “Tim Lee” at HTML text component in Fig.2, the resulted RDF document would have DC:Creator predicate as shown in Fig.3.

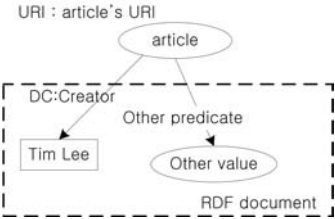


Fig. 3. Resulted RDF Document

4 System Design

4.1 Architecture

An ordinary board system is constructed using the CGI that connects articles to the backend storage. However, a semantic-based board requires additional mechanism for connecting semantic information to the backend storage. It should also support dynamic importing of a PSD described with the SBoard ontology. To satisfy these requirements, the Semantic Board is consisted of two independent modules shown in Fig.4.

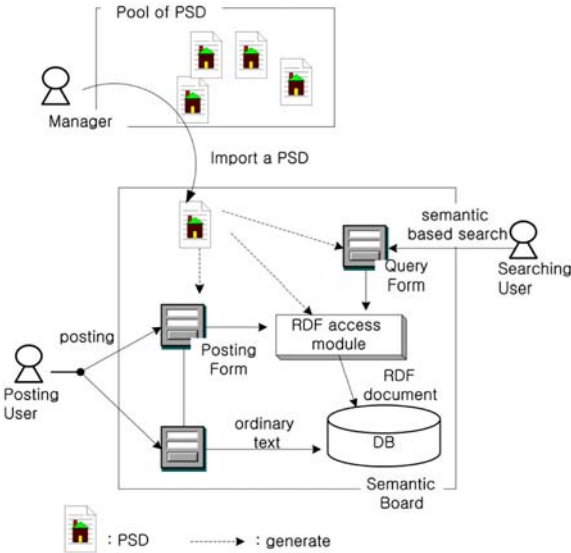


Fig. 4. Structure of the Semantic Board

One module is similar to an ordinary board system, but the other module is responsible of processing semantic information. Initially, a board manager imports a PSD. The Semantic Board reads the PSD on starting time and dynamically generates the posting/query form and RDF access module. The posting form is combined with normal article posting input form and the query form is provided as an independent page. With this scheme, a board manager is able to adopt various PSDs for supporting diverse semantic information without the board's code modification.

## 4.2 Query Statement

Board users who want to get articles by semantic searching have to use semantic query language such as RDQL [15]. The RDQL is similar to the SQL and provides searching RDF documents using a triple. Since it is difficult for ordinary users to use this, we designed query operations that generate RDQL statements in the Semantic Board. It enables users to search semantic information without using complex RDQL statement. The designed operations are as shown in Table 2.

**Table 2.** Query Operations

Data type	Operation
Numeric	GT(>), GE(>=), EQ(==), LE(<=), LT(<)
Text	SAME
Date	FROM, UNTIL

Query operations are automatically allocated to the OPTION field of SELECT component according to the data type in a PSD when the query form is generated. A *target* in the Query-Form group indicates the predicate used for searching target. The Query-Form group and RDF data group cooperate and make the query form components. For example, if there is an "age-query" predicate whose target predicate's data type is xsd:integer, the Semantic Board creates HTML SELECT form tags according to the Numeric data type as shown in Fig.5. The generated form tags can be grouped as "query operation part" and "query data part". The query operation part selects an operation to make a RDQL query operation and the query data part indicates the target data for the query operation.

When users issue a query on a query form, a selected query operation and query data are used to make corresponding RDQL statement. If a user selects "GE" as query method and enters "25" as data for age in the example Fig.5, a RDQL statement is created from the PSD and user's data input as shown in Fig.6. The Semantic Board hides complex RDQL from users adopting this scheme.

## 4.3 Evaluation

We tested the Semantic Board and the SBoard ontology by developing a job-hunting board system. Several predicates required for a job-hunting are described in a PSD,



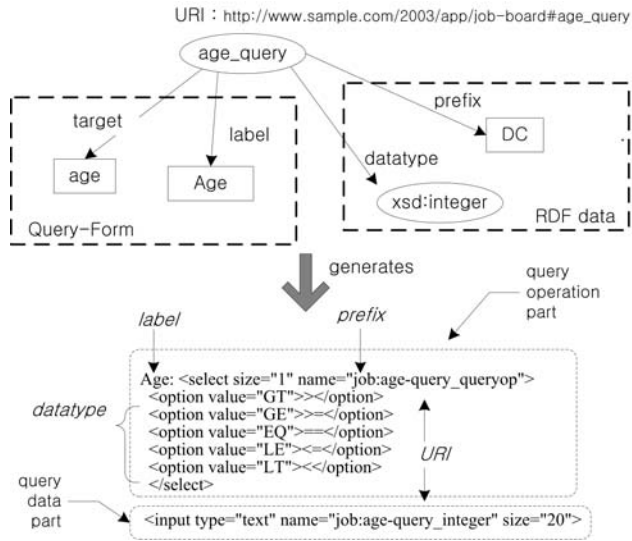


Fig. 5. Generation of Query Form

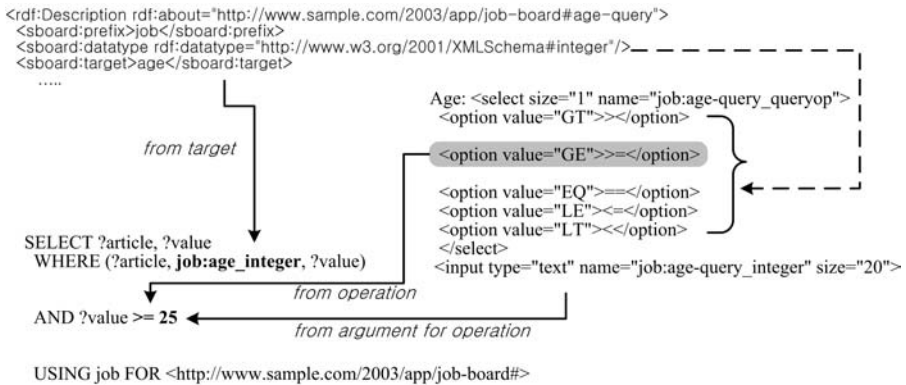


Fig. 6. Relation between PSD, Form, and RDQL statement

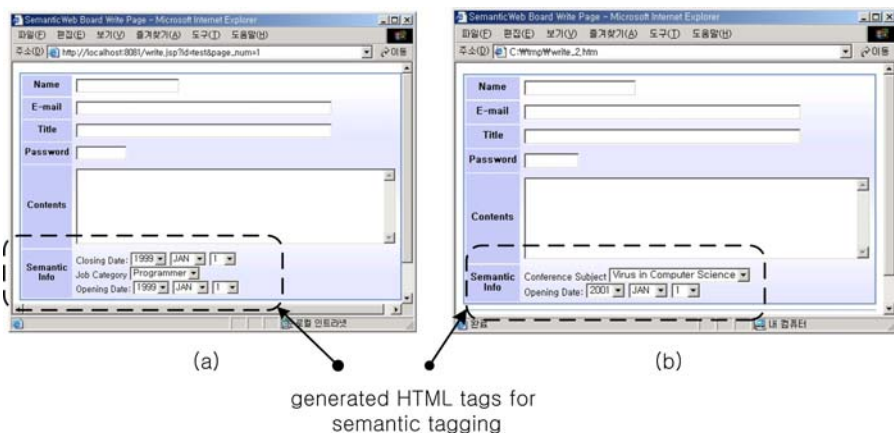
which is imported into the Semantic Board. We assumed “job category”, “opening date” and “closing date” may be important semantic information for the job-hunting board and “opening date” is targeted for searching data. The part of the PSD is as below.

The purpose of the evaluation is a testing the proper generation of posting/query forms and validity of resulted RDF document. After importing above sample job-hunting PSD, the Semantic Board generates the posting form shown in Fig.7.(a). It also enables users to access articles using semantic-based searching for “job opening date”. Besides the job-hunting PSD, other PSDs including “conference meeting” are tested on the same board. For the conference meeting PSD, users can semantically access conference information by its conference subject and opening date shown in Fig.7.(b).

```

<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xmlns:sboard="http://www.sboard.org/2003/sboard-rdf/1.0#"
  xml:base="http://www.sample.com/2003/app/job-board">
  <rdf:Description rdf:about="http://www.sboard.org/2003/model">
    <sboard:modelname>JobBoard</sboard:modelname>
  </rdf:Description>
  <rdf:Description rdf:about="http://www.sample.com/2003/app/job-board#category">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property">
    <sboard:prefix>job</sboard:prefix>
    <sboard:datatype rdf:datatype="http://www.w3.org/2001/XMLSchema#string"/>
    <sboard:label>Job Category</sboard:label>
    <sboard:formtag rdf:resource="http://www.sboard.org/2003/sboard-rdf/1.0#selection"/>
    <sboard:default>Programmer</sboard:default>
    <sboard:range rdf:resource="#Prop_001_Bag"/>
    <sboard:formtype rdf:resource="http://www.sboard.org/2003/sboard-rdf/1.0#inputform"/>
  </rdf:Description>
  <rdf:Description rdf:about="http://www.sample.com/2003/app/job-board#openingdate">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property">
    <sboard:prefix>job</sboard:prefix>
    <sboard:datatype rdf:datatype="http://www.w3.org/2001/XMLSchema#date"/>
    <sboard:formtype rdf:resource="http://www.sboard.org/2003/sboard-rdf/1.0#inputform"/>
    <sboard:label>Opening Date</sboard:label>
  </rdf:Description>
  ----- cut here -----

```



**Fig. 7.** Generated HTML Posting Form

It is difficult for ordinary users to write PSD effectively, but this is not a main obstacle because a PSD is generally written by a skilled manager or it can be easily copied from the Internet. Until now, the SBoard ontology is not able to describe the Blank Node or the Container in the RDF specification and this will be a next research issue.

## 5 Conclusion and Future Work

A bulletin board system has played an important role for a long time in exchanging information for online community, but it has revealed some weak points in accessing information. In a current board system, users have to access information at lexical level and it causes users to invest much time and effort for retrieving target data. However, this issue is not confined only to the board system, but related to most information systems.

To solve this issue, we propose a board system and a new ontology that support semantic information access. The ontology is used to describe various predicates that can indicate the semantic information of an article in the board system. The proposed board system is designed to support diverse semantic information to an article without code modification. It imports PSDs and then generates posting/query HTML forms that enable ordinary users to use semantic information easily. We also suggest a new searching mechanism to connect the RDQL and HTML form tags without data type conflict. The main purpose of this research is suggestion of a method that enables dynamic semantic information support in a board system without code modification and the evaluation shows expected results.

In the further research, the proposed system will be extended to support semantic accessing of multimedia data including pictures and an asynchronous notification based on semantic information. The predicate describing method in this paper can be used in other information systems that need to change its semantic information at runtime.

## References

1. Brisaboa, N.R., Duran, M.J., Penabad, M.R., Places, A.S.: A collaborative framework for a digital library. *Proc. CRIWG 2000* (2000) 18-20
2. Hong, H.C., Chen, Y.C.: Design and implementation of a Web-based bulletin system for official documents. *Proc.COMPSAC-2000*. (2000) 60-65
3. Nagai,M., Shiraki,K., Kato,H.,Akahori,K.: The effectiveness of a Web bulletin board enhanced with a knowledge map. *Proc. Computers in Education*. (2002) 268 - 272
4. Hyunmo, K., Shneiderman, B., Wolff, G.J.: Dynamic layout management in a multimedia bulletin board. *Proc.IEEE 2002 Symposia on Human Centric Computing Languages and Environments*. (2002) 51 -53
5. David, G., David, A.N., Brian, .M.O., Douglas.B.T.: Using collaborative filtering to weave an information tapestry. *CACM*. Vol.35. No.12. (1999) 61-70
6. Tirr, H.: Search in vain, challenges for Internet search. *IEEE Computer*. Vol.36. No.1. (2003) 115-116
7. Berners-Lee,T., Hendler, J., Lassila,O.: The Semantic Web. *Scientific American*. Vol.284. No.5. (2001) 34-43
8. Heflin, J., Hendler, J.: A Portrait of the Semantic Web in Action. *IEEE Intelligent Systems*. Vol.16. No.2. (2001) 54-59
9. Hendler, J.: Agents and the Semantic Web. *IEEE Intelligent Systems*. Vol.16. No.2. (2001) 30-37

10. Huhns, M.N.: Agents as Web Services. *IEEE Internet Computing*. Vol.6. No.4 (2002) 93-95
11. RDF Home Page: <http://www.w3.org/RDF/> (2003)
12. Decker, S., Melnik, S., van Harmelen, F., Fensel, D., Klein, M., Broekstra, J., Erdmann, M., Horrocks, I.: The Semantic Web: the roles of XML and RDF. *Internet Computing*. IEEE. Vol.4. No.5. (2000) 63 – 73
13. Noy, N.F., Sintek, M., Decker, S., Crubezy, M., Fergerson, R.W., Musen, M.A.: Creating Semantic Web contents with Protege-2000. *IEEE Intelligent Systems*. Vol.16. No.2. (2001) 60-71
14. Hai, Z.: Active e-Document Framework ADF: Model and Tool. *Information and Management*. Vol.41. No.1. (2003) 87-97
15. Andy, S.: A Programmer's Introduction to RDQL. <http://www.hpl.hp.com/semweb/doc/tutorial/RDQL/> (2002)
16. Diane, H.: Using Dublin Core. <http://dublincore.org/documents/2003/08/26/usageguide/> (2003)

# Efficient Evaluation of Sparse Data Cubes

Lixin Fu

Division of Computer Science  
Department of Mathematical Sciences  
University of North Carolina, Greensboro  
383 Bryan Building, P. O. Box 26170  
Greensboro, NC 27402-6170, U.S.A.  
lfu@uncg.edu

**Abstract.** Computing data cubes requires the aggregation of measures over arbitrary combinations of dimensions in a data set. Efficient data cube evaluation remains challenging because of the potentially very large sizes of input datasets (e.g., in the data warehousing context), the well-known curse of dimensionality, and the complexity of queries that need to be supported. This paper proposes a new dynamic data structure called *SST* (Sparse Statistics Trees) and a novel, interactive, and fast cube evaluation algorithm called *CUPS* (Cubing by Pruning *SST*), which is especially well suitable for computing aggregates in cubes whose data sets are sparse. *SST* only stores the aggregations of non-empty cube cells instead of the detailed records. Furthermore, it retains in memory the dense cubes (a.k.a. iceberg cubes) whose aggregate values are above a threshold. Sparse cubes are stored on disks. This allows a fast, accurate approximation for queries. If users desire more refined answers, related sparse cubes are aggregated. *SST* is incrementally maintainable, which makes *CUPS* suitable for data warehousing and analysis of streaming data. Experiment results demonstrate the excellent performance and good scalability of our approach.

## 1 Introduction

In the past decade there has been continuous interest by the research community on data warehousing, OLAP (On-line Analytical Processing) and data cubes [5, 6, 9]. How to compute and store data cubes is of particular importance. Since most OLAP queries involve only aggregates in the form of group-bys or cube-bys instead of detailed information, we call these types of queries *cube queries*. These queries compute the aggregates (SUM, COUNT, MIN, MAX, etc.) of measures over an *arbitrary* combination of the dimensions and their hierarchical levels. For example, if a retail warehouse contains three dimensions, time, location, and product, a cube query could be “How many computers are sold in Raleigh, NC and Atlanta, GA between January and March of year 2001?”

Efficient computation of data cube is a fundamental and required function in analytical applications. It forms the basis for generating various reports and for complex

data analysis. Moreover, efficient data cube computation has important applications in designing a new breed of data mining algorithms. The new SST data structure exhibits the following important advantages over existing approaches:

- Efficiency: SST only stores aggregations (instead of detail records) and dense data cubes. After initialisation of SST by one pass of data, accurate approximation can be instantly obtained without any I/O operations.
- Interactive: users can refine their query answers or stop evaluation at any time.
- Self-indexing: the dense cubes stored in in-memory SST and the sparse cubes stored on disk are all already sorted automatically. This eliminates the large overhead of indexing the data cubes or views.
- Scalable: CUPS can deal with many dimensions and large domain sizes. It is scalable in large number of records. In addition, CUPS is easily modified to parallel algorithms.

The rest of this paper is organized as follows. The related work is summarized in section 2. In sections 3 and 4, we present the data structure of SST and the cube query processing algorithms. The results of the performance experiments are given in section 5. Concluding remarks are provided in Sec. 6.

## 2 Related Work

The literature related to OLAP and data cube is rich. To deal with the sparse data, Yihong Zhao et al. proposed the chunking method and sparse data structure for sparse chunks [20]. Sanjay Gail and Alok Choudhary proposed PARSIMONY to parallelize MOLAP for better performance [8]. Also in the MOLAP camp, CubiST (Cubing with Statistics Trees) for the first time computes and maintains all the data cubes including supercubes together in one compact data structure called statistics tree (ST) for dense data [21]. ST is a *static* data structure. Once the dimensions and their domain sizes are given, the tree configuration is determined irrespective to the contents of the records. When ST can fit into memory, CubiST is an excellent choice. But in most real world applications, the requirement that the whole ST fit into memory is unrealistic. CUPS addresses this major drawback of scalability so that it is especially scalable and suitable for sparse data. To optimise the cube queries that have constraints on arbitrary hierarchy levels, [22] selects and materializes a family of statistics trees for dense data. CUPS is interactive, which is a very attractive feature in data exploration and data streaming applications. In addition, in this paper we will address the I/O issues (e.g. paging and matching) that papers [21, 22] did not.

Materialized views are commonly used to speedup cube queries. A greedy algorithm over the lattice structure to choose views for materialization is given [12]. Other views can be computed from them on-the-fly. View maintenance problem is addressed by [14, 10, 19, 16]. Bitmap and B<sup>+</sup>-tree are two popular indexing schemes that are used by most systems. Bitmap is suitable for dimensions with small number of values. Encoded bitmap is an improvement for large domain size dimensions [7]. B<sup>+</sup>-tree is an indexing structure for dimensions with large cardinalities. B<sup>+</sup>-tree is one-dimensional structure while SST is multidimensional.

Another method is to restrict cubing only on group-bys that use `HAVING COUNT(*) > X`, where  $X$  is greater than some threshold [4]. K. Beyer and R. Ramakrishnan develop a new algorithm BUC (Bottom-Up Cubing) to solve the new min-sup-cube problem. Similar to some ideas in [17], BUC builds the CUBE bottom-up; i.e., it builds the CUBE by starting from a group-by on a single attribute, then on two attributes, and so on. BUC belongs to the ROLAP camp, focusing on the row operations e.g. sorting. External sorting and large intermediate files slow down this type of algorithms. Another major drawback of BUC is that it is not incrementally maintainable.

T. Johnson and D. Shasha [13] propose cube trees and cube forests for cubing. SST differs from cube trees in that it stores all the aggregates in the leaves and internal nodes contain special star pointers. Due to the complexity and long response times, some algorithms give a quick approximation instead of an exact answer that requires much more time. Sampling is often used in estimations [11, 3]. Vitter and Wang use wavelets to estimate aggregates for sparse data [18]. An interesting idea of relatively low cost is to refine self-tuning histograms by using feedback from query execution engine [2]. However, the loss of accuracy in this algorithm is unacceptable for high skewed data.

### 3 Sparse Statistics Trees

#### 3.1 Tree Structure

SSTs are multi-way trees with  $k+1$  levels, where  $k$  is the number of dimensions. Each level corresponds to a dimension except the last level. The root is at level 0. The leaves on the last level (i.e. level  $k$ ) contain aggregates. The internal nodes are used to direct the access paths for the queries. A node at level  $h$  ( $h = 0, 1, \dots, k-1$ ) contains (index, pointer) pairs. The indexes represent the domain values of the corresponding dimension and the pointers direct query paths to the next level. An additional special index value called *star value* represents the ALL value of dimension  $h$ ; the corresponding *star pointer* is used to direct the path to the next level for a cube query that has no constraints for dimension  $h$ .

All leaves are automatically sorted and indexed by the corresponding root-to-leaf paths. SST stores the *full cube* (i.e. containing *all* data cubes) in its leaves if it fits in memory. In the example SST of Fig. 1, there are three dimensions and the domain value indexes are labelled along their corresponding pointers. Leaves shown in boxes contain the aggregation values of the paths from the root to the leaves. Without loss of generality, in this paper we implement COUNT function. Other aggregation functions e.g. SUM needs only minor changes. Data cubes with no or few stars consume most of the storage space but are rarely queried. When main memory is exhausted, these sparse cubes will be cut off and stored on disk. In this sense, SST makes optimal usage of memory space and is used to speed up most frequent queries. Another important observation is that SST represents a *super view*, binding all the views (or arrays) with different combinations of dimensions together into one compact structure. As a MOLAP data structure, SST preserves the advantages of ROLAP in the sense that it only stores nonempty data cubes in summary tables for sparse data.

### 3.2 Dynamic Generation and Maintenance of the SST Tree

The SST is dynamically generated. While scanning, the input records are inserted into the tree one by one. Whenever a new record is inserted, starting from the root for the first dimension, we first search its index fields. If the index is already there, we simply follow its pointer to the next level node. If not, a new node will be created as its child and a new entry of (index, pointer) pair will be inserted. If necessary, a new child is also created for the star index entry. At the same time we always follow the star pointer to the next level.

We proceed recursively until level  $k-1$  is reached. The SST after inserting the first record (6, 9, 5) and the second record (20, 1, 3) is shown in Fig. 1. The newly created or accessed pointers are shown in dashed arrows. The pseudo-code of inserting one record into SST is shown in Fig. 2 by calling recursive function `insert(root, 0, record)`. To generate SST, we first create a root with an empty list and then repeatedly insert all the input records.

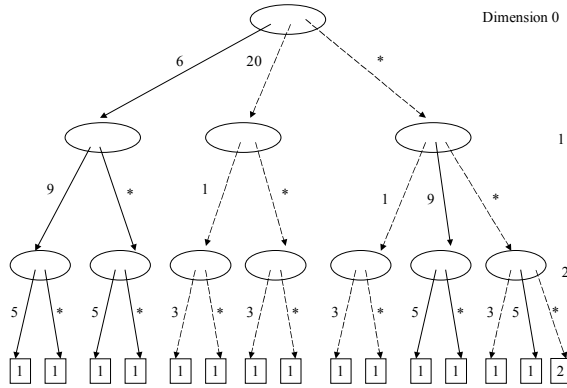


Fig. 1. The SST after inserting first two records.

Even though SST only stores the nonempty aggregates, it may still not fit in memory at some point during the insertion process. Our strategy is to cut sparse leaves off and store them on disks so that they can be retrieved later to refine answers.

### 3.3 Cutting Sparse Nodes and Generating Runs

The total number of nodes in the SST is maintained during the construction and maintenance. Once it reaches a certain threshold (we say, SST is *full*), a cut phase starts. One idea is to cut off sparse leaves whose COUNT values are less than a threshold *minSupp*.

Continuing the example of Fig. 1, two more records (6, 9, 3) and (20, 9, 3) are inserted (Fig. 3). If we set *minSupp* to 2, all the nodes and pointers shown in the dashed sub-graphs will be cut. Notice that when a node (e.g., B) has no children (i.e., has an empty list) after cut, it must also be cut and its pair entry in its parent (e.g., A) should be deleted. All memory space resulting from cut is reclaimed for later use. All the leaves being cut are stored on disk in the form of (path, aggregation) pairs, where



```
1 insert(n, level, record) { // insert a record into node n at level "level";
2   if level = k-1 then
3     // base case, where k is the number of dimensions;
4     if index is not found in the list, where index = record[level];
5       create new leaf and pointer to it; add a new entry into list;
6       create star leaf and entry if it is not there yet;
7       update the leaves pointed by index and star pointers;
8     return;
9   if index is not found in the list, where index = record[level];
10    create new node and pointer to it; add a new entry into list;
11    create start node if it is not there and add star entry into list;
12    let m1, m2 be the nodes pointed by index and star pointers;
13    insert(m1, level+1, record); insert (m2, level+1, record);
14  return;
```

Fig. 2. Recursive insertion algorithm.

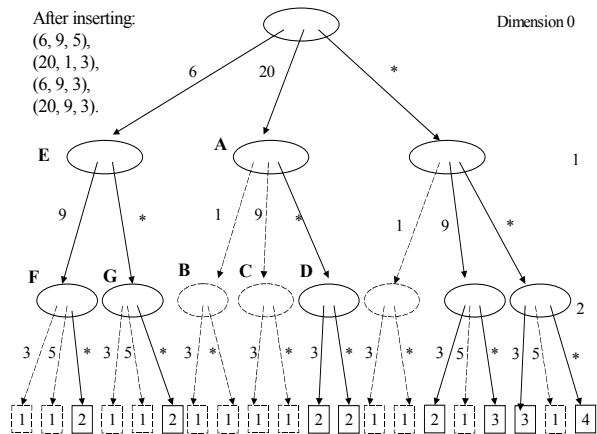


Fig. 3. SST after inserting two more records and before cutting.

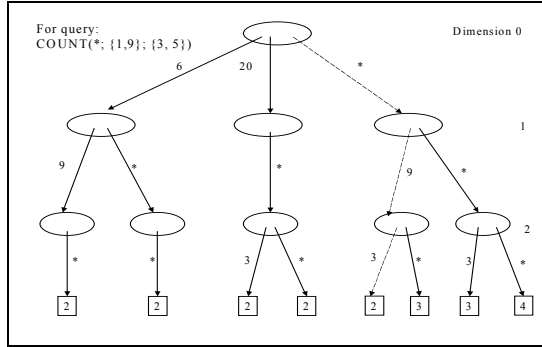
path is the root-to-leaf index combination and aggregation in our example is the COUNT value. All the pairs resulting from the same cut phase form a *run*, which are naturally sorted by paths.

## 4 Evaluation of Ad-Hoc Cube Queries

### 4.1 Approximate Query Evaluation through SST

From any cube query over  $k$  dimensions, we extract the constrained domain values and store them using  $k$  vectors  $v_0, v_1, \dots, v_{k-1}$ , each for a dimension. If dimension  $i$  is absent,  $v_i$  contains one single ALL value. We call these vectors *selected value sets*

(SVS) of the query. Starting from the root of SST, our algorithm follows along the paths directed by the SVS until it will encounter related leaves. If a path stops at a node  $n$  at level  $s$  where the  $s^{\text{th}}$  domain value is not in the index list of  $n$ , the algorithm returns 0 because there is no dense cube for this path. The aggregation of the visited leaves is returned to users as an approximation. We discuss the methods for refining query answers in the next subsection.



**Fig. 4.** The SST after the cut phase.

In our running example, suppose a cube query  $\text{COUNT}(*; \{1, 9\}; \{3, 5\})$  is submitted and the SST shown in Fig. 4 is used to evaluate the query. We first compute its SVS:  $v_0 = \{*\}$ ,  $v_1 = \{1, 9\}$ , and  $v_2 = \{3, 5\}$ . Starting from the root, follow the star pointer to the next level node. The path stops because the first domain value of  $v_1$  ( $=1$ ) is not in the list. The query result is still 0. Check the second value of  $v_2$  and follow the pointer corresponding to its index 9. Further tracking along the pointer for 3 leads to the fall-off leaf (the matched path  $*-9-3$  is shown in dashed lines). After adding it up, the result becomes 2. Following the pointer of the next value of  $v_3$  does not lead to a match. A quick approximate query result of 2 is returned. Next, we can refine the result by matching the paths with the runs generated in Sec. 3.3. The result is updated from 2 to 3 (after aggregating path  $*-1-3$  in the run), and finally to the exact answer 4 (after aggregating path  $*-9-5$ ).

## 4.2 Interactively Refining Query Answers

When the run files are large and the queries involve a large number of data cubes, retrieving and aggregating the related sparse cubes are nontrivial. The cubes are accessed by the paths defined by  $v_0 \times v_1 \times \dots \times v_{k-1}$  from SVS. We store them in a vector  $Q\_cube$ . For each run, refining query answers becomes a problem of matching the cubes in  $Q\_cube$  with the (path, value) pairs stored in the run file and aggregate those matched cubes as the refinement for this run. Of course, it is inefficient to scan the run file for matches with  $Q\_cube$ . Instead, we segment the run file into pages and only retrieve the matched pages. During the run generation, a *pageTable* is created and stored in memory. The entries of *pageTable* are the cube paths of the first cube of each page.

## 5 Simulation Results

### 5.1 Setup for Simulation

We have conducted comprehensive experiments on the performance of our CUPS algorithms by varying the number of records, and the degrees of data skew in a comparison with BUC. We decided to use BUC since the performance of BUC was better than its closest competitor MemoryCube. We have implemented BUC based on a description given in the paper [4]. However, this paper only implements the internal BUC. That is, it assumes that all the partitions are in the memory. Obviously, this is not adequate since in real world applications the space for the partitions and other intermediate results easily exceed the common available memory. Therefore, to investigate its overall behaviour including I/O operations we implement both internal BUC and external BUC. All experiments were conducted on a Dell Precision 330 with 1.7GHZ CPU, 256MB memory, and the Windows 2000 operating system.

### 5.2 Varying Number of Records

We first use uniformly distributed random data over five dimensions with cardinality of 10 each. The number of records increases from 50, 000 to 1,000,000 (data set sizes from 1megabytes to 20 megabytes). Correspondingly, the data density (defined as the average number of records per cell) increases from 0.5 to 10. The threshold *minSupp* and available memory are set to 2 and 10 megabytes respectively. The pattern of runtimes is the same for larger parameters and data sets. The runtimes are shown in Fig. 5. CUPS is about 2-4 times faster than BUC. The runtimes are the times for computing the data cubes.

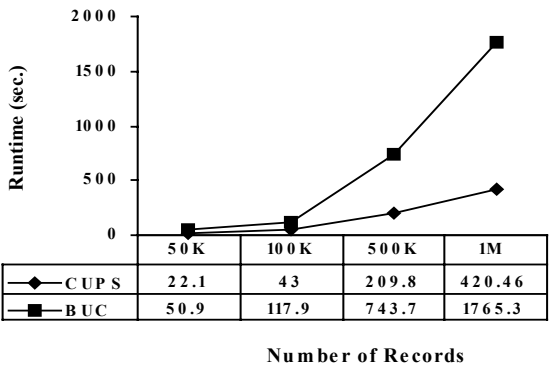


Fig. 5. Varying number of records.

### 5.3 Varying Degrees of Data Skew

In this set of experiments, instead of using uniform data sets we change the degrees of data skew. Real world data sets are often highly skewed. For example, the number of computers sold in Chicago, IL last month was probably much higher than in Greensboro, NC. Our data sets all contain 100,000 records and have 5 dimensions with the same size of 20. The average data density is  $1/32$ .

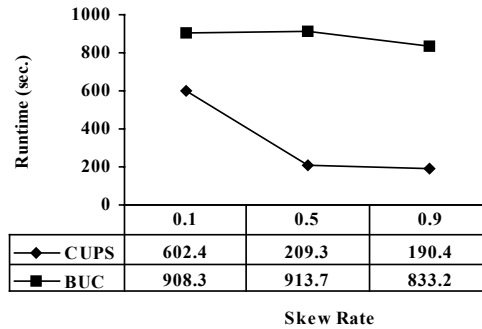


Fig. 6. Varying degrees of data skew.

To control the degree of data skew, we use a measure *skew\_rate* which is the percentage of input records that have values in  $0 - skew\_max$  for *all* dimensions, where *skew\_max* is a parameter (we fixed it to 5 in the experiments). The rest of records have random values within domain ranges (0-19 in our example). The larger *skew\_rate* and the smaller *skew\_max* are, the more skewed the data set is. The test results of using *skew\_rate* from 0.1 to 0.9 are show in Fig. 6. For very sparse and skewed data sets, the performance of CUPS is significantly better than BUC. Although the data is very sparse, some regions and subspaces are still dense.

## 6 Conclusion

In this paper, we presented a new data structure called SST. Based on SST, new cubing algorithm CUPS has been given. This method deals with data sparseness by only storing nonempty cubes and retaining only dense cubes in memory. Duplicate records are aggregated into leaves without storing the original data even the record Ids. For complex relational databases with high dimensionality and large domain sizes, to free more space, the algorithm dynamically cuts out high-dimensional sparse units that are rarely or never queried. Our optimal one-pass initialization algorithm and internal query evaluation algorithm ensure fast set-up and instant responses to the very complex cube queries. Our paging and “zigzag” matching techniques speedup the query refinement computation. Comparisons with the BUC algorithms have confirmed the benefits and good scalability of CUPS.

## References

1. S. Agarwal, R. Agrawal, P. Deshpande, J. Naughton, S. Sarawagi, and R. Ramakrishnan. On the Computation of Multidimensional Aggregates. In *Proceedings of the International Conference on Very Large Databases (VLDB'96)*, Mumbai (Bomabi), India, 1996.
2. Ashraf Aboulmaga and Surajit Chaudhuri. Self-tuning histograms: building histograms without looking at data. In *Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data (SIGMOD '99)*, Philadelphia, PA, pages 181-192, June 1999.
3. Swarup Acharya, Phillip B. Gibbons, Viswanath Poosala. Congressional samples for approximate answering of group-by queries. In *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data (SIGMOD '00)*, Dallas, TX, pages 487-498, May 2000.
4. Kevin Beyer, Raghu Ramakrishnan. Bottom-up computation of sparse and Iceberg CUBE. In *Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data (SIGMOD '99)*, Philadelphia, PA, pages 359-370, June 1999.
5. E. F. Codd, S. B. Codd, and C. T. Salley. Providing OLAP (on-line analytical processing) to user-analysts: An IT mandate. *Technical Report*, [www.arborsoft.com/OLAP.html](http://www.arborsoft.com/OLAP.html).
6. S. Chaudhuri and U. Dayal. An Overview of Data Warehousing and OLAP Technology, *SIGMOD Record*, **26**(1): 65-74, 1997.
7. C. Y. Chan and Y. E. Ioannidis. Bitmap Index Design and Evaluation. In *Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data (SIGMOD '98)*, Seattle, WA, pages 355-366, 1998.
8. Sanjay Goil, Alok N. Choudhary. PARSIMONY: An Infrastructure for Parallel Multidimensional Analysis and Data Mining. *Journal of Parallel and Distributed Computing* 61(3): 285-321, 2001.
9. J. Gray, S. Chaudhuri, A. Bosworth, A. Layman, D. Reichart, M. Venkatrao, F. Pellow, and H. Pirahesh. Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Totals, *Data Mining and Knowledge Discovery*, **1**(1): 29-53, 1997.
10. A. Gupta, V. Harinarayan, and D. Quass. Aggregate-query Processing in Data Warehousing Environments. In *Proceedings of the Eighth International Conference on Very Large Databases (VLDB '95)*, Zurich, Switzerland, pages 358-369, 1995.
11. Phillip B. Gibbons, Yossi Matias. New sampling-based summary statistics for improving approximate query answers. In *Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data (SIGMOD '98)*, Seattle, WA, pages 331-342, 1998.
12. V. Harinarayan, A. Rajaraman, and J. D. Ullman. Implementing data cubes efficiently. *SIGMOD Record (ACM Special Interest Group on Management of Data)*, **25**(2): 205-216, 1996.
13. T. Johnson and D. Shasha, "Some Approaches to Index Design for Cube Forests," *Bulletin of the Technical Committee on Data Engineering, IEEE Computer Society*, **20**:1, pp. 27-35, 1997.
14. D. Lomet, *Bulletin of the Technical Committee on Data Engineering*, **18**, IEEE Computer Society, 1995.
15. Wolfgang Lehner, Richard Sidle, Hamid Pirahesh, Roberta Wolfgang Cochrane. Maintenance of cube automatic summary tables. In *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data (SIGMOD '00)*, Dallas, TX, pages 512-513, May 2000.

16. Inderpal Singh Mumick, Dallan Quass, Barinderpal Singh Mumick: Maintenance of Data Cubes and Summary Tables in a Warehouse. In *Proceedings of the 1997 ACM SIGMOD International Conference on Management of Data (SIGMOD '97)*, Tucson, AZ, pages 100-111, 1997.
17. Ramakrishnan Srikant, Rakesh Agrawal. Mining quantitative association rules in large relational tables. In *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data (SIGMOD '96)*, Montreal, Canada, pages 1-12, 1996.
18. Jeffrey Scott Vitter, Min Wang. Approximate computation of multidimensional aggregates of sparse data using wavelets. In *Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data (SIGMOD '99)*, Philadelphia, PA, pages 193-204, June 1999.
19. W. P. Yan and P. Larson. Eager Aggregation and Lazy Aggregation. In *Proceedings of the Eighth International Conference on Very Large Databases (VLDB '95)*, Zurich, Switzerland, pages 345-357, 1995.
20. Y. Zhao, P. M. Deshpande, and J. F. Naughton. An Array-Based Algorithm for Simultaneous Multidimensional Aggregates. *SIGMOD Record (ACM Special Interest Group on Management of Data)*, **26**(2): 159-170, 1997.
21. Fu, L. and Hammer, J. CUBIST: A New Algorithm For Improving the Performance of Ad-hoc OLAP Queries. in *ACM Third International Workshop on Data Warehousing and OLAP, Washington, D.C, USA, November, 2000*, 72-79.
22. Hammer, J. and Fu, L. Improving the Performance of OLAP Queries Using Families of Statistics Trees. in *3rd International Conference on Data Warehousing and Knowledge Discovery DaWaK 01, September, 2001, Munich, Germany, 2001*, 274-283.

# Redundancy Free Mappings from Relations to XML

Millist W. Vincent, Jixue Liu, and Chengfei Liu

School of Computer and Information Science  
University of South Australia  
{millist.vincent,jixue.liu,chengfei.liu}@unisa.edu.au

**Abstract.** Given the fact that relational and object-relational databases are the most widely used technology for storing data and that XML is the standard format used in electronic data interchange, the process of converting relational data to XML documents is one that occurs frequently. The problem that we address in this paper is an important one related to this process. If we convert a relation to an XML document, under what circumstances is the XML document redundancy free? In some allied work we formally defined functional dependencies in XML (XFDs) and, based on this definition, formally defined redundancy in an XML document. We then introduced a normal form for an XML document (XNF) and showed that it is a necessary and sufficient condition for the elimination of redundancy. In this paper we address the problem of determining what class of mappings map a relation in BCNF to an XML document in XNF. The class of mappings we consider is very general and allows arbitrary nesting of the original flat relation. Our main result establishes a necessary and sufficient condition on the DTD induced by the mapping for it to be in XNF.

## 1 Introduction

The eXtensible Markup Language (XML) [4] has recently emerged as a standard for data representation and interchange on the Internet [19, 1]. As a result of this and the fact that relational and object-relational databases are the standard technology in commercial applications, the issue of converting relational data to XML data is one that frequently occurs. In this conversion process of relational data to XML data, there are many different ways that relational data can be mapped to XML data, especially considering the flexible nesting structures that XML allows. This gives rise to the following important problem. Are some mappings ‘better’ than others?

Firstly, one has to make precise what is meant by ‘better’. In this paper we extend the classical approach used in relational database design and regard a mapping as good if it produces an XML document which is free of redundancy. This then raises the question of what is precisely meant by redundancy in an XML document. The relationship between normal forms and redundancy elimination has been investigated, both for the relational case [11, 7, 10] and the nested

relational case [8], and in particular it has been shown that *Boyce-Codd normal form* (BCNF) [6] is a necessary and sufficient condition for the elimination of redundancy in relations when the only constraints are *functional dependencies* (FDs). In some recent work [18, 12, 17], we showed how to extend the definition of FDs in relations to FDs in XML (*called XFDs*) and how to extend the definition of redundancy from relations to XML. In particular we defined a normal form for a set of XFDs (XNF) and showed that it is a necessary and sufficient condition for every document satisfying the set of XFDs to be free of redundancy.

In a previous paper [18] we addressed the following problem. Suppose we are given a single relation and wish to map it to an XML document. There are many such mappings and in particular a deeply nested structure, rather than a flat structure, may be chosen because it better represents the semantics of the data. We then want to determine what mappings result in the XML document being redundancy free. Knowing this is important for systems designers because they would obviously wish to avoid mappings which result in the introduction of redundancy to the XML document since, as shown in [18], they can lead to update problems which parallel those that occur in unnormalized flat relations. The class of mappings that we considered is a very general class of mappings from a relation into an XML document first proposed in [12, 12]. The class takes a relation, first converts it into a nested relation by allowing an *arbitrary* sequence of nest operations and then converts the nested relation into an XML document. This is a very general class of mappings and we believe that it covers all the types of mappings that are likely to occur in practice. The main result of the paper then showed that, for the case where all FDs in the relation are unary, any mapping from the general class of mappings from a relation to an XML document will always be redundancy free if and only if the relation is in BCNF.

However, there are certain limitations to this previous work that we address in this paper. In particular, in [18] we were only concerned with eliminating redundancy from those XML documents which are the result of mappings from normalised relations, rather than all possible XML documents defined over the schema.

As a result, it is possible that an XML document mapped from a relation to be redundancy free even though the set of XFDs applying to the document is not in XNF. While this is fine if the document is to remain unchanged, it is not satisfactory if the XML document is to be subsequently updated because the fact that the document is not in XNF means that redundancy can arise even if the XFDs in the document are checked after an update. As a result of this discussion, in this paper we address the following problem. Given a relation scheme which is in BCNF, and a set of unary keys, what mappings from a relation defined over a scheme to an XML document result in the set of XFDs induced by the mapping being in XNF? If a mapping has this property, then because of the result on the relationship between XNF and redundancy elimination mentioned earlier, one is guaranteed that any XML document will be free of redundancy, even if later updated, as long as the XFDs are checked after an update. The main result of the paper (Theorem 3) derives a necessary and sufficient condition on the



DTD induced by the mapping for it to be in XNF. Not surprising, because we are placing more stringent requirements on the mapping from relations to XML documents, Theorem 3 shows that only a very limited class of mappings produce normalized documents whereas the result in [18] showed that any mapping will result in a redundancy free document as long as the relation scheme is in BCNF.

The rest of this paper is organized as follows. Section 2 contains some preliminary definitions. Section 3 contains the definition of XFDs. Section 4 addresses the topic of how to map flat relations to XML documents and establishes one of the main results of the paper concerning the XFDs induced by the mappings. Section 5 addresses the problem of how to determine when a mapping from a relation to an XML document results in the document being in XNF. In Section 5 we derive the main result of the paper which provides a characterization in terms of the induced DTD. Finally, Section 6 contains some concluding remarks.

## 2 Preliminary Definitions

In this section we present some preliminary definitions that we need before defining XFDs. We model an XML document as a tree as follows.

**Definition 1.** Assume a countably infinite set  $\mathbf{E}$  of element labels (tags), a countable infinite set  $\mathbf{A}$  of attribute names and a symbol  $\mathbf{S}$  indicating text. An XML tree is defined to be  $T = (V, lab, ele, att, val, v_r)$  where  $V$  is a finite set of nodes in  $T$ ;  $lab$  is a function from  $V$  to  $\mathbf{E} \cup \mathbf{A} \cup \{\mathbf{S}\}$ ;  $ele$  is a partial function from  $V$  to a sequence of  $V$  nodes such that for any  $v \in V$ , if  $ele(v)$  is defined then  $lab(v) \in \mathbf{E}$ ;  $att$  is a partial function from  $V \times \mathbf{A}$  to  $V$  such that for any  $v \in V$  and  $l \in \mathbf{A}$ , if  $att(v, l) = v_1$  then  $lab(v) \in \mathbf{E}$  and  $lab(v_1) = l$ ;  $val$  is a function such that for any node in  $v \in V$ ,  $val(v) = v$  if  $lab(v) \in \mathbf{E}$  and  $val(v)$  is a string if either  $lab(v) = \mathbf{S}$  or  $lab(v) \in \mathbf{A}$ ;  $v_r$  is a distinguished node in  $V$  called the root of  $T$  and we define  $lab(v_r) = root$ . Since node identifiers are unique, a consequence of the definition of  $val$  is that if  $v_1 \in \mathbf{E}$  and  $v_2 \in \mathbf{E}$  and  $v_1 \neq v_2$  then  $val(v_1) \neq val(v_2)$ . We also extend the definition of  $val$  to sets of nodes and if  $V_1 \subseteq V$ , then  $val(V_1)$  is the set defined by  $val(V_1) = \{val(v) | v \in V_1\}$ .

For any  $v \in V$ , if  $ele(v)$  is defined then the nodes in  $ele(v)$  are called subelements of  $v$ . For any  $l \in \mathbf{A}$ , if  $att(v, l) = v_1$  then  $v_1$  is called an attribute of  $v$ . Note that an XML tree  $T$  must be a tree. Since  $T$  is a tree the set of ancestors of a node  $v$ , is denoted by  $Ancestor(v)$ . The children of a node  $v$  are also defined as in Definition 1 and we denote the parent of a node  $v$  by  $Parent(v)$ . of  $Ancestor$ .

We note that our definition of  $val$  differs slightly from that in [5] since we have extended the definition of the  $val$  function so that it is also defined on element nodes. The reason for this is that we want to include in our definition paths that do not end at leaf nodes, and when we do this we want to compare element nodes by node identity, i.e. node equality, but when we compare attribute or text nodes we want to compare them by their contents, i.e. value equality. This point will become clearer in the examples and definitions that follow.

We now give some preliminary definitions related to paths.

**Definition 2.** A path is an expression of the form  $l_1 \cdots l_n$ ,  $n \geq 1$ , where  $l_i \in \mathbf{E} \cup \mathbf{A} \cup \{S\}$  for all  $i$ ,  $1 \leq i \leq n$  and  $l_1 = \text{root}$ . If  $p$  is the path  $l_1 \cdots l_n$  then  $\text{Last}(p) = l_n$ .

For instance, if  $\mathbf{E} = \{\text{root}, \text{Division}, \text{Employee}\}$  and  $\mathbf{A} = \{D\#, \text{Emp}\# \}$  then  $\text{root}$ ,  $\text{root.Division}$ ,  $\text{root.Division.D}\#$ , and  $\text{root.Division.Employee.Emp}\#.S$  are all paths.

**Definition 3.** Let  $p$  denote the path  $l_1 \cdots l_n$ . The function  $\text{Parnt}(p)$  is the path  $l_1 \cdots l_{n-1}$ . Let  $p$  denote the path  $l_1 \cdots l_n$  and let  $q$  denote the path  $q_1 \cdots q_m$ . The path  $p$  is said to be a prefix of the path  $q$ , denoted by  $p \subseteq q$ , if  $n \leq m$  and  $l_1 = q_1, \dots, l_n = q_n$ . Two paths  $p$  and  $q$  are equal, denoted by  $p = q$ , if  $p$  is a prefix of  $q$  and  $q$  is a prefix of  $p$ . The path  $p$  is said to be a strict prefix of  $q$ , denoted by  $p \subset q$ , if  $p$  is a prefix of  $q$  and  $p \neq q$ . We also define the intersection of two paths  $p_1$  and  $p_2$ , denoted by  $p_1 \cap p_2$ , to be the maximal common prefix of both paths. It is clear that the intersection of two paths is also a path.

For example, if  $\mathbf{E} = \{\text{root}, \text{Division}, \text{Employee}\}$  and  $\mathbf{A} = \{D\#, \text{Emp}\# \}$  then  $\text{root.Division}$  is a strict prefix of  $\text{root.Division.Employee}$  and  $\text{root.Division.D}\# \cap \text{root.Division.Employee.Emp}\#.S = \text{root.Division}$ .

**Definition 4.** A path instance in an XML tree  $T$  is a sequence  $\bar{v}_1 \cdots \bar{v}_n$  such that  $\bar{v}_1 = v_r$  and for all  $\bar{v}_i$ ,  $1 < i \leq n$ ,  $v_i \in V$  and  $\bar{v}_i$  is a child of  $\bar{v}_{i-1}$ . A path instance  $\bar{v}_1 \cdots \bar{v}_n$  is said to be defined over the path  $l_1 \cdots l_n$  if for all  $\bar{v}_i$ ,  $1 \leq i \leq n$ ,  $\text{lab}(\bar{v}_i) = l_i$ . Two path instances  $\bar{v}_1 \cdots \bar{v}_n$  and  $\bar{v}'_1 \cdots \bar{v}'_m$  are said to be distinct if  $v_i \neq v'_i$  for some  $i$ ,  $1 \leq i \leq n$ . The path instance  $\bar{v}_1 \cdots \bar{v}_n$  is said to be a prefix of  $\bar{v}'_1 \cdots \bar{v}'_m$  if  $n \leq m$  and  $\bar{v}_i = \bar{v}'_i$  for all  $i$ ,  $1 \leq i \leq n$ . The path instance  $\bar{v}_1 \cdots \bar{v}_n$  is said to be a strict prefix of  $\bar{v}'_1 \cdots \bar{v}'_m$  if  $n < m$  and  $\bar{v}_i = \bar{v}'_i$  for all  $i$ ,  $1 \leq i \leq n$ . The set of path instances over a path  $p$  in a tree  $T$  is denoted by  $\text{Paths}(p)$ .

For example, in Figure 1,  $v_r.v_1.v_3$  is a path instance defined over the path  $\text{root.Division.Section}$  and  $v_r.v_1.v_3$  is a strict prefix of  $v_r.v_1.v_3.v_4$ .

We now assume the existence of a set of legal paths  $P$  for an XML application. Essentially,  $P$  defines the semantics of an XML application in the same way that a set of relational schema define the semantics of a relational application.  $P$  may be derived from the DTD, if one exists, or  $P$  be derived from some other source which understands the semantics of the application if no DTD exists. The advantage of assuming the existence of a set of paths, rather than a DTD, is that it allows for a greater degree of generality since having an XML tree conforming to a set of paths is much less restrictive than having it conform to a DTD. Firstly we place the following restriction on the set of paths.

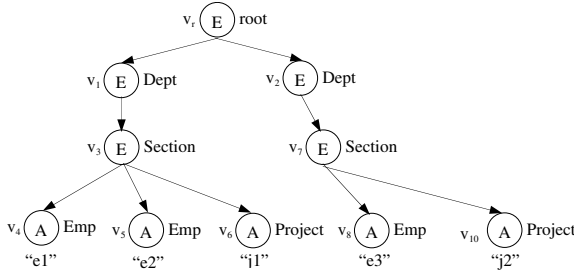
**Definition 5.** A set  $P$  of paths is consistent if for any path  $p \in P$ , if  $p_1 \subset p$  then  $p_1 \in P$ .

This is natural restriction on the set of paths and any set of paths that is generated from a DTD will be consistent.

We now define the notion of an XML tree conforming to a set of paths  $P$ .

**Definition 6.** Let  $P$  be a consistent set of paths and let  $T$  be an XML tree. Then  $T$  is said to conform to  $P$  if every path instance in  $T$  is a path instance over  $\text{apath}$  in  $P$ .

The next issue that arises in developing the machinery to define XFDs is the issue is that of missing information. This is addressed in [18] but in this paper, because of space limitations, we take the simplifying assumption that there is no missing information in XML trees. More formally, we have the following definition.



**Fig. 1.** A complete XML tree.

**Definition 7.** Let  $P$  be a consistent set of paths, let  $T$  be an XML that conforms to  $P$ . Then  $T$  is defined to be complete if whenever there exist paths  $p_1$  and  $p_2$  in  $P$  such that  $p_1 \subset p_2$  and there exists a path instance  $\bar{v}_1 \dots \bar{v}_n$  defined over  $p_1$ , in  $T$ , then there exists a path instance  $\bar{v}'_1 \dots \bar{v}'_m$  defined over  $p_2$  in  $T$  such that  $\bar{v}_1 \dots \bar{v}_n$  is a prefix of the instance  $\bar{v}'_1 \dots \bar{v}'_m$ .

For example, if we take  $P$  to be  $\{\text{root}, \text{root.Dept}, \text{root.Dept.Section}, \text{root.Dept.Section.Emp}, \text{root.Dept.Section.Project}\}$  then the tree in Figure 1 conforms to  $P$  and is complete.

The next function returns all the final nodes of the path instances of a path  $p$  in  $T$ .

**Definition 8.** Let  $P$  be a consistent set of paths, let  $T$  be an XML tree that conforms to  $P$ . The function  $N(p)$ , where  $p \in P$ , is the set of nodes defined by  $N(p) = \{\bar{v} | \bar{v}_1 \dots \bar{v}_n \in \text{Paths}(p) \wedge \bar{v} = \bar{v}_n\}$ .

For example, in Figure 1,  $N(\text{root.Dept}) = \{v_1, v_2\}$ .

We now need to define a function that returns a node and its ancestors.

**Definition 9.** Let  $P$  be a consistent set of paths, let  $T$  be an XML tree that conforms to  $P$ . The function  $\text{AAncessor}(v)$ , where  $v \in V$ , is the set of nodes in  $T$  defined by  $\text{AAncessor}(v) = v \cup \text{Ancestor}(v)$ .

For example in Figure 1,  $\text{AAncessor}(v_3) = \{v_r, v_1, v_3\}$ . The next function returns all nodes that are the final nodes of path instances of  $p$  and are descendants of  $v$ .

**Definition 10.** Let  $P$  be a consistent set of paths, let  $T$  be an XML tree that conforms to  $P$ . The function  $Nodes(v, p)$ , where  $v \in V$  and  $p \in P$ , is the set of nodes in  $T$  defined by  $Nodes(v, p) = \{x | x \in N(p) \wedge v \in AAncestor(x)\}$ .

For example, in Figure 1,  $Nodes(v_1, root.Dept.Section.Emp) = \{v_4, v_5\}$ . We also define a partial ordering on the set of nodes as follows.

**Definition 11.** The partial ordering  $>$  on the set of nodes  $V$  in an XML tree  $T$  is defined by  $v_1 > v_2$  iff  $v_2 \in Ancestor(v_1)$ .

### 3 Strong Functional Dependencies in XML

We recall the definition of an XFD from [18, 12].

**Definition 12.** Let  $P$  be a set of consistent paths and let  $T$  be an XML tree that conforms to  $P$ . An XML functional dependency (XFD) is a statement of the form:  $p_1, \dots, p_k \rightarrow q$ ,  $k \geq 1$ , where  $p_1, \dots, p_k$  and  $q$  are paths in  $P$ .  $T$  strongly satisfies the XFD if  $p_i = q$  for some  $i$ ,  $1 \leq i \leq k$  or for any two distinct path instances  $\bar{v}_1, \dots, \bar{v}_n$  and  $\bar{v}'_1, \dots, \bar{v}'_n$  in  $Paths(q)$  in  $M(T)$ ,  $val(\bar{v}_n) \neq val(\bar{v}'_n) \Rightarrow \exists i, 1 \leq i \leq k$ , such that  $x_i \neq y_i$  if  $Last(p_i) \in \mathbf{E}$  else  $\perp \notin Nodes(x_i, p_i)$  and  $\perp \notin Nodes(y_i, p_i)$  and  $val(Nodes(x_i, p_i)) \cap val(Nodes(y_i, p_i)) = \emptyset$ , where  $x_i = \max\{v | v \in \{\bar{v}_1, \dots, \bar{v}_n\} \wedge v \in N(p_i \cap q)\}$  and  $y_i = \max\{v | v \in \{\bar{v}'_1, \dots, \bar{v}'_n\} \wedge v \in N(p_i \cap q)\}$ .

We note that since the path  $p_i \cap q$  is a prefix of  $q$ , there exists only one node in  $\bar{v}_1, \dots, \bar{v}_n$  that is also in  $N(p_i \cap q)$  and so  $x_i$  is always defined and unique. Similarly for  $y_i$ .

### 4 Mapping from relations to XML

We assume that the reader is familiar with the definition of the nest operator,  $\nu_Y(r^*)$ , and the unnest operator,  $\mu_{\{Y\}}(r^*)$ , for nested relations as defined in [9, 3].

The translation of a relation into an XML tree consists of two phases. In the first we map the relation to a nested relation whose nesting structure is arbitrary and then we map the nested relation to an XML tree.

In the first step we let the nested relation  $r^*$  be defined by  $r_i = \nu_{Y_{i-1}}(r_{i-1})$ ,  $r_0 = r$ ,  $r^* = r_n$ ,  $1 \leq i \leq n$  where  $r$  represents the initial (flat) relation and  $r^*$  represents the final nested relation. The  $Y_i$  are allowed to be arbitrary, apart from the obvious restriction that  $Y_i$  is an element of the NRS for  $r_i$ .

In the second step of the mapping procedure we take the nested relation and convert it to an XML tree as follows. We start with an initially empty tree. For each tuple  $t$  in  $r^*$  we first create an element node of type  $Id$  and then for each  $A \in Z(N(r^*))$  we insert a single attribute node with a value  $t[A]$ . We then repeat recursively the procedure for each subtuple of  $t$ . The final step in the procedure is to compress the tree by removing all the nodes containing nulls from the tree. We now illustrate these steps by an example.

Name	Sid	Major	Class	Exam	Proj
Anna	Sid1	Maths	CS100	Mid	Proj A
Anna	Sid1	Maths	CS100	Mid	Proj B
Anna	Sid1	Maths	CS100	Final	Proj A
Anna	Sid1	Maths	CS100	Final	Proj B

Fig. 2. A flat relation.

Example 1. Consider the flat relation shown in Figure 2.

If we then transform the relation  $r$  in Figure 2 by the sequence of nestings  $r_1 = \nu_{PROJ}(r)$ ,  $r_2 = \nu_{EXAM}(r_1)$ ,  $r_3 = \nu_{CLASS,\{EXAM\},\{PROJ\}}(r_2)$ ,  $r^* = \nu_{MAJOR}(r_3)$  then the relation  $r^*$  is shown in Figure 3. We then transform the nested relation in Figure 3 to the XML tree shown in Figure 4.

Name	Sid	{Major}	{Class {Exam}}	{Proj}}
Anna	Sid1	Maths	CS100	Mid
				Final
				Proj A
				Proj B

Fig. 3. A nested relation derived from a flat relation.

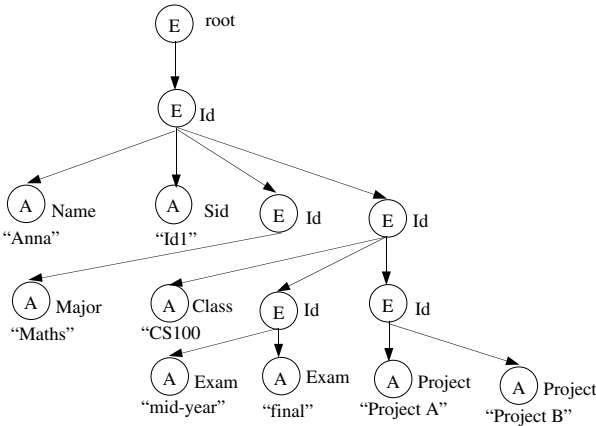


Fig. 4. An XML tree derived from a nested relation.

We now have the major result which establishes the correspondence between satisfaction of FDs in relations and satisfaction of XFDs in XML. We denote by  $T_{r^*}$  the XML tree derived from  $r^*$ . We refer to the XFDs satisfied in  $T_{r^*}$  as the *induced XFDs*.

**Theorem 1.** *Let  $r$  be a flat relation and let  $A \rightarrow B$  be an FD defined over  $r$ . Then  $r$  strongly satisfies  $A \rightarrow B$  iff  $T_{r^*}$  strongly satisfies  $p_A \rightarrow p_B$  where  $p_A$  denotes the path in  $T_{r^*}$  to reach  $A$  and  $p_B$  denotes the path to reach  $B$ .*

In addition the mapping induces a DTD for  $T_{r^*}$ , what we refer to as the *induced DTD*. To describe this we firstly formally define a restricted class of DTDs which is sufficient for the purpose of this paper.

**Definition 13.** *A DTD is a specification of the form  $D ::= e|(e(D_1, \dots, D_n))^*$  where  $e \in \mathbf{E}$  and  $D_1, \dots, D_n$  are DTDs and  $*$  represents repetition one or more times.*

For example, in the previous example the induced DTD is:

$$D ::= (Id(Name, Sid, (Id(Major))^*, (Id(Class, (Id(Exam))^*, (Id(Proj))^*))^*))^*.$$

## 5 Mapping from Normalized Relations to Normalized XML Documents

Firstly, we recall the definition of the normal form XNF from [2, 18].

**Definition 14.** *Let  $P$  be a consistent set of paths and let  $\Sigma$  be a set of XFDs such that  $P_\Sigma \subseteq P$ .  $\Sigma$  of XFDs is in XML normal form (XNF) if for every nontrivial XFD  $p_1, \dots, p_k \rightarrow q \in \Sigma^+$ ,  $Last(q) \notin \mathcal{S}$  and if  $Last(q) \in \mathbf{A}$  then  $p_1, \dots, p_k \rightarrow Parnt(q) \in \Sigma^+$ , where  $\Sigma^+$  denotes the set of XFDs logically implied by  $\Sigma$ .*

This leads to the following important result which was established in [18].

**Theorem 2.** *Let  $P$  be a consistent set of paths and let  $\Sigma$  be a set of XFDs such that  $P_\Sigma \subseteq P$ . Then  $\Sigma$  does not cause redundancy iff  $\Sigma$  is in XNF.*

Before proceeding to the main result of the paper, we firstly address the issue of what XFDs to apply to the XML document from a mapping as given in Section 4. As already shown in Theorem 1, the only XFDs induced by a mapping from a relation to an XML document are those of the form  $p_A \rightarrow p_B$  corresponding to an FD  $A \rightarrow B$ . However, if these are the only XFDs that we assume apply to the XML document then no XML document will be in XNF. So what we propose is to add an extra XFD to the set of induced XFDs to ensure that this pathological situation does not occur. The XFD that we add is the combination of every path terminating in an attribute node determines the *root.Id* node in the tree. This ensures that there are no duplicate subtrees rooted at the highest *Id* node. In general, we have the following definition.

**Definition 15.** *Let  $R(A_1, \dots, A_n)$  denote a relation scheme which is in BCNF, let  $\Sigma_R$  denote a set of unary FDs defined over  $R$ . Let  $\Omega$  be the set of all mappings from relations defined over  $R$  to XML trees as defined in Section 4 and let  $\omega \in \Omega$ . Let  $\Sigma_\omega$  be the set of XFDs as defined by Theorem 1. Then the set of XFDs induced by  $\omega$ , denoted by  $\Sigma_{i_\omega}$ , is defined by  $\Sigma_{i_\omega} = \Sigma_\omega \cup \{p_{A_1}, \dots, p_{A_n} \rightarrow root.Id\}$ .*

We now turn to the main topic of this paper, determining when  $\Sigma_{i_\omega}$  is in XNF. One approach would be to use the definition of XNF directly since the implication problem for unary XFDs has been shown to be decidable and a linear time algorithm has been developed for it [15]. In other words, we simply proceed

by taking each XFD  $p \rightarrow q \in \Sigma_{i_\omega}^+$  and using the implication algorithm in [15] to determine if  $p \rightarrow \text{Parnt}(q) \in \Sigma_{i_\omega}^+$ . However this approach neither provides any insight into the properties of those mappings which result in XML trees in XNF, nor is it easy to use. It requires that the system designer be familiar both with the theory of XFDs and the implication algorithm for them. Instead, what we do now is a much more efficient and simpler approach. Instead of using the induced XFDs directly, we shall characterize when  $\Sigma_{i_\omega}$  is in XNF by the structure of the induced DTD. This result is much easier to apply as it requires only the knowledge of the keys in the relation scheme and the structure of the DTD; it does not require any understanding of XFDs. Thus we now have the main result of this paper.

**Theorem 3.** *Let  $R(A_1, \dots, A_n)$  denote a relation scheme which is in BCNF and let  $\Sigma_R$  denote a set of unary FDs defined over  $R$ . Let  $D_\omega$  be the DTD induced by  $\omega$ . Then  $\Sigma_{i_\omega}$  is in XNF iff the following hold:*

- (i) *if  $R$  has more than one key then  $D_\omega$  has the form  $D_\omega ::= (\text{Id}(A_1, \dots, A_n))^*$ ;*
- (ii) *If  $R$  has only one key, call it  $A_1$ , then  $D_\omega$  has the form  $D_\omega ::= (\text{Id}(A_1, \dots, A_n))^*$  or  $D_\omega ::= (\text{Id}(A_2, \dots, A_n, (\text{Id}(A_1))^*))^*$ .*

## 6 Conclusions

The problem that we have addressed in this paper is one related to this process of exporting relational data in XML format. The problem is that if one converts a relation to an XML document, under what circumstances will the XML document be normalized, i.e. be in XNF as defined in [18]. The XML document being in XNF is important since, as shown in [18], it is a necessary and sufficient condition from the elimination of redundancy in an XML document. Being redundancy free is an important property of an XML document since, as shown in [18], it guarantees the absence of certain types of update anomalies in the same fashion that redundancy elimination and BCNF ensures the elimination of update anomalies in relational databases [11].

The mappings that we allow from relations to XML documents were first introduced in [12] and are very general. Essentially, they allow arbitrary nesting of a flat relation into an XML document and we believe that the class considered includes all the mappings that are likely to occur in practice. Drawing on some previous work by the authors [12, 18, 17] that formally defined functional dependencies and redundancy in XML documents, we firstly showed that FDs in flat relation map to XFDs in XML documents in a natural fashion, i.e. if an FD holds in a flat relation if and only if a related XFD holds in the mapped XML document. The main result of the paper (Theorem 3) then gave a necessary and sufficient condition on the DTD induced by the mapping for the XML document to be in XNF.

There are several other related issues that we intend to investigate in the future. The first is to relax the assumption that the exported relation is in BCNF. In general, the relation to be exported will be a materialized view of

the source database and so, even if the relations in the source database are normalized, the materialized will not necessarily be normalized. We need then to derive a similar result to the one derived in this paper to determine when an unnormalized flat relation is mapped to an XML document. The second issue is to consider the same problem for the case where the constraints in the source relation are MVDs. In some allied work [14, 13, 16], we have shown how to extend the definition of multivalued dependencies to XML documents and have defined a 4NF for XML (4XNF) and shown it to be a sufficient condition for the elimination of redundancy. Thus an important problem to be addressed is determining which mappings from a relation to XML are in 4XNF. The last issue that we intend to address is the issue of mappings between XML documents, rather than between relations and XML documents as considered in this paper. In this case the class of mappings will be different and may be those that can be expressed in a language such as Xquery. The problem then to be addressed is which mappings produce normalized XML documents.

## References

1. S. Abiteboul, P. Buneman, and D. Suciu. Data on the Web. Morgan Kaufman, 2000.
2. M. Arenas and L. Libkin. A normal form for XML documents. In Proc. ACM PODS Conference, pages 85–96, 2002.
3. P. Atzeni and V. DeAntonellis. Foundations of databases. Benjamin Cummings, 1993.
4. T. Bray, J. Paoli, and C.M. Sperberg-McQueen. Extensible markup language (XML) 1.0. Technical report, <http://www.w3.org/Tr/1998/REC-XML-19980819>, 1998.
5. P. Buneman, S. Davidson, W. Fan, and C. Hara. Reasoning about keys for XML. In International Workshop on Database Programming Languages, 2001.
6. E.F. Codd. Recent investigations in relational database systems. In IFIP Conference, pages 1017–1021, 1974.
7. M. Levene and M. W. Vincent. Justification for inclusion dependency normal form. IEEE Transactions on Knowledge and Data Engineering, 12:281–291, 2000.
8. W.Y. Mok, Y.K. Ng, and D. Embley. A normal form for precisely characterizing redundancy in nested relations. ACM Transactions on Database Systems, 21(1):77–106, 1996.
9. S.J. Thomas and P.C. Fischer. Nested relational structures. In P. Kanellakis, editor, The theory of databases, pages 269–307. JAI Press, 1986.
10. M. W. Vincent. A new redundancy free normal form for relational database design. In B. Thalheim and L. Libkin, editors, Database Semantics, pages 247–264. Springer Verlag, 1998.
11. M. W. Vincent. Semantic foundations of 4NF in relational database design. Acta Informatica, 36:1–41, 1999.
12. M.W. Vincent and J. Liu. Functional dependencies for XML. In Fifth Asian Pacific Web Conference, 2003.
13. M.W. Vincent and J. Liu. Multivalued dependencies and a 4NF for XML. In 15th International Conference on Advanced Information Systems Engineering (CAISE), pages 14–29. Lecture Notes in Computer Science 2681 Springer, 2003.



14. M.W. Vincent and J. Liu. Multivalued dependencies in XML. In 20th British National Conference on Databases (BNCOD), pages 4–18. Lecture Notes in Computer Science 2712 Springer, 2003.
15. M.W. Vincent, J. Liu, and C. Liu. The implication problem for unary functional dependencies in XML. In submitted for publication, 2003.
16. M.W. Vincent, J. Liu, and C. Liu. Multivalued dependencies and a redundancy free 4NF for XML. In International XML database symposium (Xsym), pages 254–266. Lecture Notes in Computer Science 2824 Springer, 2003.
17. M.W. Vincent, J. Liu, and C. Liu. Strong functional dependencies and a redundancy free normal form for XML. In 7th World Multi-Conference on Systemics, Cybernetics and Informatics, 2003.
18. M.W. Vincent, J. Liu, and C. Liu. Strong functional dependencies and their application to normal forms in XML. Accepted for publication, ACM Transactions on Database Systems, February 2004, 2004.
19. J. Widom. Data management for XML – research directions. IEEE data Engineering Bulletin, 22(3):44–52, 1999.

# Tree Multivalued Dependencies for XML Datasets\*

Lawrence V. Saxton and Xiquan Tang

Department of Computer Science  
University of Regina, Canada  
{saxton, tang112x}@cs.uregina.ca

**Abstract.** This paper introduces tree multivalued dependencies (TMVDs) for XML documents by extending multivalued dependencies in relational databases and recent research about tree functional dependencies (TFDs). We show the existence of the TMVDs in XML documents that produce redundancies and update anomalies. Secondly, the precise definition of TMVDs is proposed, which is based on the tree structure and deep equality. Moreover, we introduce the concept of recomposition that is a reconstruction of an XML database by moving or adding nodes. The right side of TMVDs is characterized by single attribute and multiple attributes, for which there are different recompositions. In addition, the relationship between TFDs and TMVDs is investigated, and we demonstrate that a TMVD is a generalized TFD. Furthermore, we present a theorem showing the recomposition by TMVDs saves space. Finally, we exhibit a recomposition that is not equivalent to any relational data.

## 1 Introduction

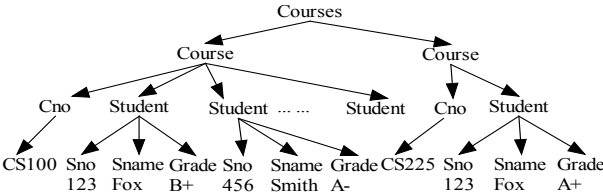
XML has recently emerged as the new universal data format for representing and exchanging data [20]. It becomes even more important as web databases are generated increasingly. As a more complex data model than the relational database model, XML documents suffer from data redundancies and update anomalies [1,20]. Consequently, storage facilities and update anomalies of XML are significant for designing well-organized XML data.

For relational databases, normalization is a well-developed technique for reducing redundancies and eliminating update anomalies [8, 4, 5, 14]. Normalization allows the decomposition of a relational schema into different sub-schemas. The recent work on TFDs [1, 2] extended the notion of functional dependencies (FDs) in a relational database to XML trees, and has been shown to be useful in reducing update anomalies and redundancies in XML.

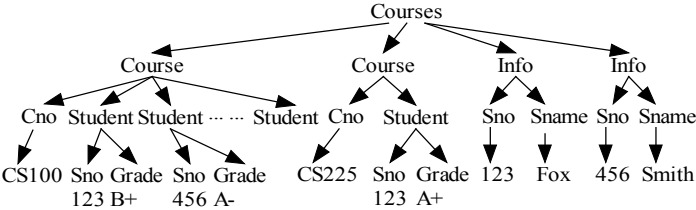
For example, Figure 1 shows an XML document that obeys the TFD  $Sno \rightarrow Sname$ , which intuitively means any two students with the same value of *Sno* must have the same *Sname*. Moreover, the transformation of Figure 1 is presented in Figure 2 by moving attributes *Sname* and creating new element types *Info*, and thus reducing redundancies and anomalies.

---

\* This research was made possible through funding from the Natural Sciences and Engineering Research Council of Canada.



**Fig. 1.** An XML document with TFD Sno → Sname

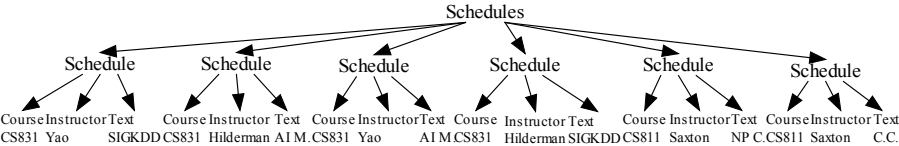


**Fig. 2.** Recomposed XML tree of Figure 1

In relational databases, multivalued dependencies (MVDs) are a generalization of FDs. In this paper, we extend TFDs in XML databases and MVDs in relational databases, and introduce tree multivalued dependencies (TMVDs). TMVDs are analogous to MVDs in relational databases. Figure 3 shows an XML document with TMVD Course →t→ Instructor, and a complete definition of TMVD appears in section 2.

Consider the following DTD that describes a part of a university data set;

```
<! ELEMENT Schedules (Schedule*) >
<! ELEMENT Schedule (Course, Instructor, Text) >
<! ELEMENT Course (#PCDATA) >
<! ELEMENT Instructor (#PCDATA) >
<! ELEMENT Text (#PCDATA) >
```



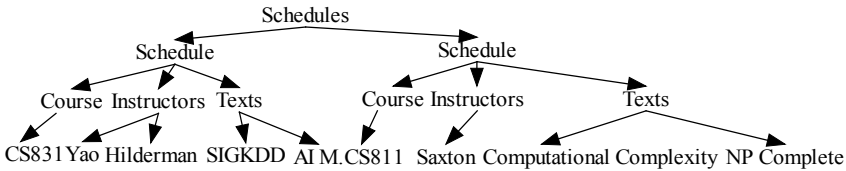
**Fig. 3.** An XML document with TMVD Course →t→ Instructor

For every Schedule, we store its Course, Teacher, and Text. Figure 3 is an XML document satisfying this DTD; we handle an XML schema in the similar way as a DTD. This XML document conforms to the following constraint. For any course there can exist any number of instructors and texts, instructors and texts are independent. In other words, the same texts are used for any given offering, no matter which instructor is offering it. A given instructor or given text can be associated with any number of courses. This example is similar to MVDs in relational databases. However, we refer to both the values and the structures of XML databases.

The TMVD constraint in Figure 3 generates redundant information in an XML document. For instance, two Instructors and two Texts have been combined for the Course “CS831” and the cross product of those items is shown in the tree. In addition, the redundancies can cause update anomalies. For example, if a new Text should be inserted for Course “CS831”, the updating has to combine two Instructors according to the constraint rather than producing only one Schedule.

We apply a method parallel to the relational database method to reduce redundancies and updating anomalies. The XML tree is reconstructed by splitting information about Course, Instructors, and Text, as shown in Figure 4, which satisfies the following DTD

```
<! ELEMENT Schedules (Schedule*) >
<! ELEMENT Schedule (Course, Instructors*, Texts*) >
<! ELEMENT Course (#PCDATA) >
<! ELEMENT Instructors (#PCDATA) >
<! ELEMENT Texts (#PCDATA) >
```



**Fig. 4.** Recomposed XML tree of Figure 2

Each Schedule element has a child as one course, and a sequence of any number of instructors and texts joined to that course. Each course is related to the number of corresponding instructors and texts. Figure 4 shows a reconstructed XML document that conforms to this DTD.

In section 2 we precisely define the concept of TMVDs. We also introduce the essential notations for TMVDs: tree, subtree, minimum subtree and deep equality. In section 3 we illustrate the definition of TMVDs by an example.

In section 4 we define the normalization of XML documents to decrease the redundancy and update anomalies in XML documents. We show the recomposition for single attribute XML documents, and multiple attributes XML documents independently. In section 5 we compare the normalizing of TFDs and TMVDs, and use a few examples to show TMVDs are a generalized TFDs.

In section 6 we present a theorem showing that the recomposition of the XML documents saves space over the original XML document. In section 7 we introduce a TFD that is introduced in [1], which is not equivalent to a relational FD. We show that it is really one of the recompositions of TMVDs, and TMVDs can be hold in this non-relational tree.

## 2 Notation

The structure of XML is fundamentally tree oriented [16, 19]. We view XML documents as trees, with nodes labelled with tag or attribute names, or atomic values [19].

To introduce the notion of tree multivalued dependencies (TMVDs), we would like to present XML trees as sets of subtrees which contain the minimum amount of information. We introduce the concept of minimum subtrees to include sufficient TMVDs information with the least space, and use deep equality to describe the equality of the subtrees.

**Definition 1 Tree.** A *tree* is defined to be a connected graph without cycles in which there is an unique path from the root to any node.

**Definition 2 Subtree.** A *subtree*  $T'$  of tree  $T$  is a tree having all of its nodes in  $T$ .

**Definition 3 Deep Equality.** Two XML subtrees are said to be *deep equal* if

1. They are atomic and the values are equal, or
2. They are sets of the same cardinality and the elements of each set are pairwise deep equal.

With the definition of the tree, subtree, and deep equality, we extend MVDs in relational databases and tree functional dependencies [1] to tree multivalued dependencies (TMVDs) in the XML documents. We define TMVDs for XML documents by using subtrees. For a DTD  $D$ , a TMVD over  $D$  is an expression of form  $X \rightarrow t \rightarrow Y$ .

**Definition 4 Tree Multivalued Dependencies.** (TMVDs)  $X \rightarrow t \rightarrow Y$

Given a DTD  $D$ , where  $X, Y$  are finite non-empty subtrees of a XML tree  $T$ ,

A XML tree  $T$  can be tested for satisfaction of  $X \rightarrow t \rightarrow Y$  if:

1.  $X \rightarrow t \rightarrow Y$  if for every pair finite non-empty subtrees  $u$  and  $v$  are subtrees of  $T$ ,  $X$  and  $Y$  are subtrees of  $u$  and  $v$ , where  $u.X$  is the subtree  $u$  including subtree  $X$
2.  $u.X$  is deep-equal to  $v.X$ ,  $u.X$  and  $u.Y$  are not deep-equal there exists  
 $z$ , a finite non-empty subtree of  $T$ , and  $X, Y$  and  $Z$  are subtrees of  $u$  and  $v$   
 where  $U$  is a minimum subtree including  $u.X, u.Y$ , and  $u.Z$ ,  $V$  is a minimum subtree including  $v.X, v.Y$ , and  $v.Z$   
 $w$ , a finite non-empty subtree of  $T$  and  $Z$  is a subtree of  $w$ .  $X, Y$  and  $Z$  are subtrees of  $w$ , where  $W$  is a minimum subtree including  $w.X, w.Y$ , and  $w.Z$   
 With  $w.X$  is deep-equal to  $u.X$ ,  $w.Y$  is deep-equal to  $u.Y$ , and  $w.Z$  is deep-equal to  $v.Z$ .

### 3 Example of TMVDs

We illustrate the definition of TMVDs by the example in Figure 3, and the TMVD:  $\text{Course} \rightarrow t \rightarrow \text{Instructor}$  exists.

**X:** Schedules.Schedule.Course"CS831"

**Y:** Schedules.Schedule.Instructor"Yao",  
Schedules.Schedule.Instructor"Hilderman"

**Z:** Schedules.Schedule.Text"SIGKDD", Schedules.Schedule.Text"AI M."

Let  $u$  include  $u.X$  Schedules.Schedule.Course"CS831",  $u.Y$  Schedules.Schedule.Instructor"Yao" and  $u.Z$  Schedules.Schedule.Text"SIGKDD"

Let  $v$  include  $v.X$  Schedules.Schedule.Course"CS831",  $v.Y$  Schedules.Schedule.Instructor"Hilderman" and  $v.Z$  Schedules.Schedule.Text"AI M."

Let  $w$  include  $w.X$  Schedules.Schedule.Course"CS831",  $w.Y$  Schedules.Schedule.Instructor"Yao" and  $w.Z$  Schedules.Schedule.Text"AI M."  
 $w.X$  is deep-equal to  $u.X$ ,  $w.Y$  is deep-equal to  $u.Y$ , and  $w.Z$  is deep-equal to  $v.Z$ .

## 4 Example of Recomposition for TMVDs

With the definitions of the previous sections, we can present the reconstruction of the trees to save space and reduce anomalies. In this section, we show the example of recomposition for single attributes and multiple attributes for TMVDs, respectively.

In BCNF, relational databases are decomposed into sub-schemas in which every nontrivial functional dependency defines a key. TFDs use a similar technique as the relational one; however, since TFDs only deal with one XML document, it actually creates a new element type and moves an attribute [1]. Instead of decomposing the tree, we propose the notion of recomposition, which is the reconstruction of the tree by moving or adding the nodes.

**Single Attribute.** One unique Course in Figure 3 is associated with a number of Instructors and Texts, and the join of Instructors and Texts creates the redundancies. Figure 4 is a recomposition by  $\text{Course} \rightarrow_t \text{Instructor} \mid \text{Text}$ , where Instructor and Text only contain a single element.

**Multiple Attributes.** The XML document in Figure 5 shows for any Course there can exist any number of Instructors and Texts, which satisfies the following DTD. Instructor includes FName and LName instead of a single element. And FName and LName should be ensured as the brother nodes.

```
<! ELEMENT Schedules (Schedule*) >
<! ELEMENT Schedule (Course, FName, LName, Texts) >
<! ELEMENT Course (#PCDATA) >
<! ELEMENT FName (#PCDATA) >
<! ELEMENT LName (#PCDATA) >
<! ELEMENT Texts (#PCDATA) >
```

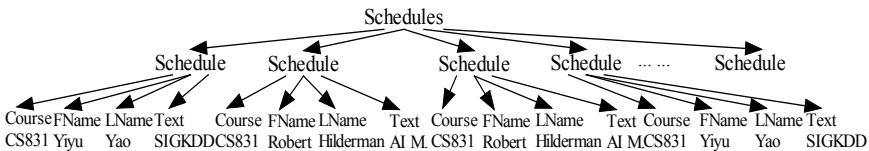


Fig. 5. An XML document with multiple attributes for TMVDs

Based on the decomposition of relational database, a good recomposition of this XML data can be generated by moving attributes, changing attribute Instructor to Instructors, and creating an extra layer Instructor for FName and LName. The extra layer Instructor has a semantic meaning, and is necessary for the structure of XML document. It ensures FName and LName as the children of Instructor, and thus retains them together.

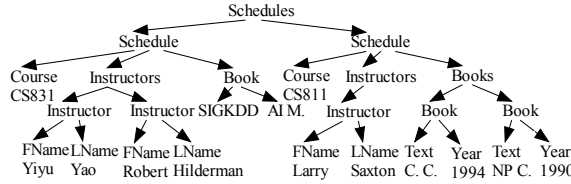


Fig. 6. Recomposition of XML document with multiple attributes for TMVDs

## 5 TMVDs and TFDs

In this section we apply the recomposition rule of TFDs for the XML document with constraint TMVDs. We display the relationship between the TFDs and TMVDs. TFDs is a special case of TMVDs, as FDs and MVDs in relational database.

**Applying TMVDs Recomposition on TFDs.** We recompose the Figure 1 as it has TMVD constraint.

Figure 1 originally has the constraint  $Sno \rightarrow Sname$ : Two students with the same Sno value must have the same Sname. Now we recompose it as it has  $Sno \rightarrow t \rightarrow Sname$ , as Sno has a number of Sname and Null. And we have Figure 4 that is the same result as Figure 2. Therefore, TFDs can be treated as same as TMVDs, and TFDs is a one case of TMVDs.

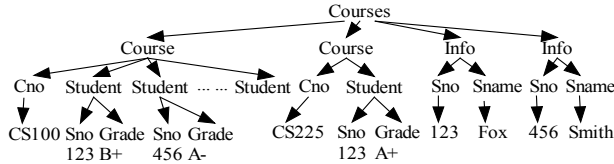


Fig. 7. Applying TMVDs recomposition on TFDs

**Applying TMVDs Recomposition on TFDs.** We apply the recomposition rule of TFDs for Figure 3

Figure 3 originally has a TMVD:  $Course \rightarrow t \rightarrow Instructor$ : for any course there can exist any number of instructors and texts. Now we recompose it as it has  $Course \rightarrow Instructor$ , as the same Course value must have the same Instructor. We have Figure 8 that is not the same result as Figure 4. Accordingly, we can't treat TMVDs the same as TFDs, and TFDs is a special case of TMVDs.

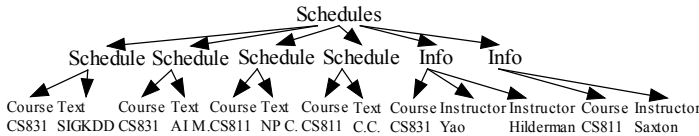


Fig. 8. Applying TMVDs recomposition on TFDs

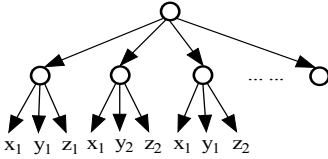
## 6 Recomposition Theorem

**Theorem 1.** The recomposition of XML documents that contain TMVDs requires no more space than the original XML documents, and contains the same information.

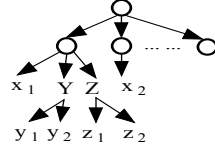
**Proof:**

**Single Attribute**

1. If  $X \rightarrow t \rightarrow Y|Z$ ,  $Y$  and  $Z$  only has a single attribute respectively,



**Fig. 9.** A single attribute XML tree



**Fig. 10.** Recomposition of Figure 9

According to the definition of TMVD, there are at least 9 leaves in the original XML tree:  $u.X$ ,  $u.Y$ ,  $u.Z$ ,  $v.X$ ,  $v.Y$ ,  $v.Z$ ,  $w.X$ ,  $w.Y$ ,  $w.Z$ . Figure 9 has at least 9 leaves:  $x_1$ ,  $y_1$ ,  $z_1$ ,  $x_1$ ,  $y_2$ ,  $z_2$ ,  $x_1$ ,  $y_1$ ,  $z_2$ . In the thorough example Figure 3, we at least have

1.  $t(\text{Schedules.Schedule.@Course}) = \text{CS831}$
2.  $t(\text{Schedules.Schedule.@Instructor}) = \text{Yao}$
3.  $t(\text{Schedules.Schedule.@Text}) = \text{SGIKDD}$
4.  $t(\text{Schedules.Schedule.@Course}) = \text{CS831}$
5.  $t(\text{Schedules.Schedule.@Instructor}) = \text{Hilderman}$
6.  $t(\text{Schedules.Schedule.@Text}) = \text{AI Magazine}$
7.  $t(\text{Schedules.Schedule.@Course}) = \text{CS831}$
8.  $t(\text{Schedules.Schedule.@Instructor}) = \text{Hilderman}$
9.  $t(\text{Schedules.Schedule.@Text}) = \text{SGIKDD}$

9 leaves, which is same to the number in the definition.

According to the definition of TMVDs, " $u.X$  is deep-equal to  $v.X$ ,  $u.X$  and  $u.Y$  are not deep-equal", and " $w.X$  is deep-equal to  $u.X$ ,  $w.Y$  is deep-equal to  $u.Y$ , and  $w.Z$  is deep-equal to  $v.Z$ ", we at least have 5 identical leaves:  $u.X$ ,  $u.Y$ ,  $u.Z$ ,  $v.Y$ ,  $v.Z$ . If  $u.Y = v.Y$ ,  $u.Z = v.Z$ , then we have a TFD instead. Figure 10 also has at least 5 identical leaves:  $x_1$ ,  $y_1$ ,  $y_2$ ,  $z_2$ ,  $z_2$ , as same as the number from the definition.

In Figure 4, we at least have 5 leaves

1.  $t(\text{Schedules.Schedule.@Course}) = \text{CS831}$
2.  $t(\text{Schedules.Schedule.Instructor}) = \text{Yao}$
3.  $t(\text{Schedules.Schedule.Instructor}) = \text{Hilderman}$
4.  $t(\text{Schedules.Schedule.Text}) = \text{SGIKDD}$
5.  $t(\text{Schedules.Schedule.Text}) = \text{AI Magazine}$

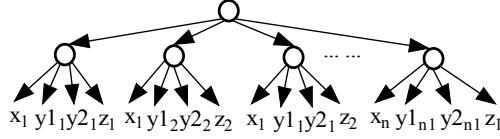
It is obvious that the original design uses more space than this recomposed tree, since the information is only stored once for both instructors.

### Multiple Attributes

We suppose a relational XML tree  $T$  satisfies  $X \rightarrow t \rightarrow Y$ , and  $Y = \{y_1, y_2, \dots, y_n\}$ . We separate the original tree  $T$  to two parts, a minimum subtree  $T'$  include  $u$ ,  $v$ ,  $w$ , and the rest. We only consider a minimum subtree  $T'$ .



**Step 1.** Suppose  $n=2$ , then  $Y = \{y1, y2\}$



**Fig. 11.** Original XML tree T with Y has two attributes y1 and y2

According to the definition of TMVD, there are at least  $3*(X+Y+Z)=3*(1+n+1) = 3*(1+2+1)=12$  nodes in the original XML tree

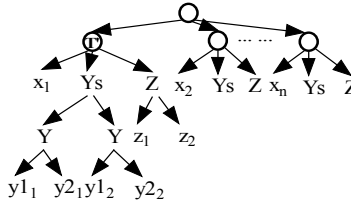
u.X, u.y1, u.y2, u.Z

v.X, v.y1, v.y2, v.Z

w.X, w.y1, w.y2, w.Z

T' in Figure 11 has at least  $(x_1+y1_1+y2_1+ z_1+x_1+y1_2+y2_2+z_2+x_1+y1_1+y2_1+z_2) = 12$  nodes in original tree, which is as same as the number from the definition.

A recomposition of Figure 11 is shown as the following as Figure 12



**Fig. 12.** Recomposed XML Tree with Y has two attributes y1 and y2

According to the definition of TMVDs, "u.X is deep-equal to v.X , u.X and u.Y are not deep-equal", and "w.X is deep-equal to u.X , w.Y is deep-equal to u.Y, and w.Z is deep-equal to v.Z", we have at least u.X, u.y1, u.y2, u.Z, v.y1, v.y2, v.Z

$$(u.X + u.y1 + u.y2 + u.Z + v.y1 + v.y2 + v.Z)$$

$$= (X + 2*y1 + 2*y2 + 2*Z)$$

$$= (X + 2*(y1+y2) + 2*Z):$$

$$(1 + 2*n + 2) = 7$$

identical nodes for any recomposed trees, and those nodes could be subtrees.

Figure 12 have at least

$$(x_1+y1_1+y2_1+y1_2+y2_2+z_1+z_2)$$

$$= (X + 2*y1 + 2*y2 + 2*Z)$$

$$= (X + 2*(y1+y2) + 2*Z) :$$

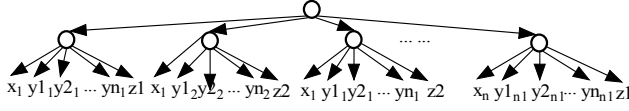
$$(1 + 2*n + 2) = 7$$

identical nodes and those nodes are subtrees, which is as same as the number of nodes from definition.

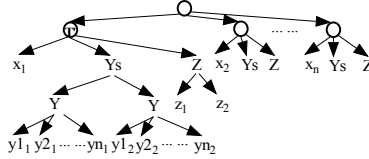
Figure 12 T' has 7 nodes, and any other recomposed tree should have at least 7 nodes. Therefore, Figure 7 costs no more space than the original tree.

An extra layer is created in the recomposition when Y has a set of attributes. Node Y in original tree is changed to Ys, and Ys has at least two children or attributes named Y, shown as Figure 14.

**Step 2.** When  $Y$  has  $n$  attributes, then  $Y = \{y_1, y_2, \dots, y_n\}$



**Fig. 13.** Original XML Tree  $T$  - sets attributes



**Fig. 14.** Recomposed Tree with sets attributes

From the definition the recomposed tree has

$$\begin{aligned}
 & u.X, u.y_1, u.y_2, \dots, u.y_n, u.Z, v.y_1, v.y_2, \dots, v.y_n, v.Z \\
 & (u.X + u.y_1 + u.y_2 + \dots + u.y_n + u.Z + v.y_1 + v.y_2 + \dots + v.y_n + v.Z) \\
 & = (u.X + (u+v).y_1 + (u+v).y_2 + \dots + (u+v).y_n + (u+v).Z) \\
 & = (u.X + (u+v).(y_1 + y_2 + \dots + y_n) + (u+v).Z) : \\
 & (1 + 2*n + 2) = 2n+3
 \end{aligned}$$

identical nodes and those nodes are subtrees, and they are at least no more than the original tree.

Figure 14 has

$$\begin{aligned}
 & (x_1 + y_1 + y_2 + \dots + y_n + y_1 + y_2 + \dots + y_n + z_1 + z_2) \\
 & = (x_1 + (y_1 + y_2) + (y_2 + y_2) + \dots + (y_n + y_n) + (z_1 + z_2)) \\
 & = (x_1 + 2*y_1 + 2*y_2 + \dots + 2*y_n + (z_1 + z_2)) \\
 & = (X + 2*(y_1 + y_2 + \dots + y_n) + 2*Z) : \\
 & (1 + 2*n + 2) = 2n+3
 \end{aligned}$$

identical nodes and those nodes are subtrees as well, which is the same as the number of nodes from the definition.

**Step 3.** If  $Z$  has sets of attributes, or both  $Y$  and  $Z$  has sets of attributes, they have the similar recomposition and proof as step 2.

## 7 Non-relational XML Document

The previous work reference about TFDs gave an example of non-relational XML document, which has two @years in the different levels of the tree, and thus causes anomalies [1]. The following algorithm shows how to detect anomalies of those kinds. We can transform documents in a lossless fashion into ones that do not yield those problems by recomposition, which is presented by figure 16.

```

for (i=0; i<length of the tree; i++)
{select a node  $a_i$ 
  for (j=i; j<length of the tree; j++)
    {compare with another node  $a_{(i+j)}$ 
      if ( $a_i \neq a_{(i+j)}$ )
        go to node  $a_{(i+j+1)}$ 
      else
        if (the length of  $a_i \neq$  the length of  $a_{(i+j)}$ )
          {non-rel; //a non-relational XML tree
//The While loop checks deep equality for a non-
relational XML tree
        {while (the children of  $a_i$ ) is not a leave
          compare (the children of  $a_i$ ) and (the children
of  $a_{(i+j)}$ ))
          {if (not equal)
            break;
          else
            if (the length of  $a_i \neq$  the length of  $a_{(i+j)}$ )
              non-rel; // a non-relational XML tree
with deep equality
            else
              break;}}}}

```

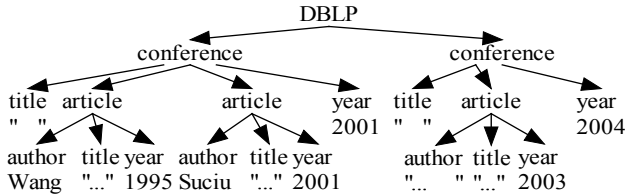


Fig. 15. A Non-relational XML Document for TFDs

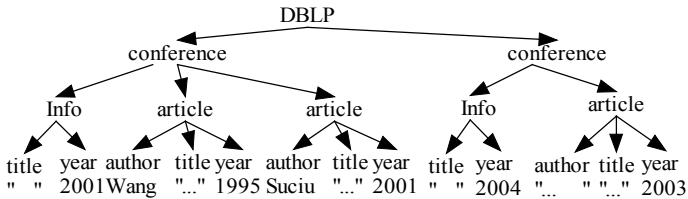


Fig. 16. Recomposed XML tree

## 8 Conclusions and Future Research

This paper presents the TMVDs of XML documents, and uses the recomposition to normalize XML data and achieve a good design. We propose the definition of TMVDs for XML documents, and show that TFDs is a special case of TMVDs. And the recomposition is able to transform the non-relational XML trees with TFDs into a relational structure.

The recomposition algorithm can be improved in several ways, and we plan to work on the order of recomposition. In addition, we also like to find the relationship between TMVDs and other dependencies. To accomplish this, the long-range goal of this research is to develop the theory of normalization for XML documents.

## References

1. Marcelo Arenas and Leonid Libkin. A Normal Form for XML Documents. In Proceedings of the 21st ACM SIGACT-SIGMOD-SIGART, Madison, USA, pages 85-96. 2002.
2. Marcelo Arenas and Leonid Libkin. An Information-Theoretic Approach to Normal Forms for Relational and XML Data. In Proceedings of the 22nd ACM SIGACT-SIGMOD-SIGART, San Diego, USA, pages 15-26. 2003.
3. William W. Armstrong and Claude Delobel, Decompositions and Functional Dependencies in Relations. ACM TODS Vol. 5 No. 4, pages 404-430, December 1980.
4. Catriel Beeri, Philip A. Bernstein, Computational Problems Related to the Design of Normal Form Relational Schemas, ACM TODS Vol. 4, No.1, pages 30-59, March 1979.
5. Catriel Beeri, On the Membership Problem for Functional and Multivalued Dependencies in Relational Databases. ACM TODS Vol. 5, No. 3, pages 241-259, September 1980.
6. Berge, Claude. Graphs and Hypergraphs. North-holland publishing company, 1973
7. Chris J. Date. An Introduction to Database Systems. Addison Wesley, 7<sup>th</sup> edition, 2000.
8. Ronald Fagin, Multivalued Dependencies and a New Normal Form for Relational Databases. ACM TODS Vol. 2, No. 3, pages 262-278, 1977.
9. Usama Fayyad and Ramasamy Uthurusamy. Data Mining and Knowledge Discovery in Databases. Communication of the ACM, Vol. 39, No. 9, pages 24-25, November 1996.
10. Mary Fernandez, Morishima Atsuyuki, Dan Suciu, and Tan Wang-Chiew. Publishing Relational Data in XML: the SilkRoute Approach. IEEE Data Engineering Bulletin, 24(2), 2001.
11. Michael Goebel and Le Gruenwald, A Survey of Data Mining and Knowledge Discovery Software Tools, SIGKDD Explorations, Vol.1, No. 1, pages 20-33, 1999.
12. Michael T. Goodrich and Roberto Tamassia. Data Structures and Algorithms in Java. John Wiley and Sons Inc., 2001.
13. Frank Harary, Graph Theory, Addison-Wesley, pages 32-42.
14. David Maier, The Theory of Relational Databases. Computer Science Press, Rockville, Maryland, 1983.
15. Leanne. M. Seaward, L.V. Saxton, Measuring Changes in Streaming XML Documents, In Sixth Joint Conference on Information Sciences, pages 232-234, March 2002.
16. Dan Suciu. The XML Typechecking Problem. SIGMOD Record, Vol. 31, No. 1, pages 89-96, 2002.
17. Limsoon Wong. Normal Forms and Conservative Properties for Query Languages over Collection Types. In Proceedings of the 12th ACM Symposium on Principles of Database Systems, Washington, DC, pages 26-36, May 1993.
18. Carlo Zaniolo, Michel A. Melkanoff. On the Design of Relational Database Schemata. ACM TODS Vol. 6, No. 1, pages 1-47, 1981.
19. W3C and XML. <http://www.w3c.org/xml>
20. Extensible Markup Language. <http://www.w3.org/TR/REC-xml>
21. XML Query. <http://www.w3.org/XML/Query>

# TROS: A Novel Model and Its Implementation for XML Data Management\*

Wenwen Qi<sup>1,2</sup> and Aoying Zhou<sup>1</sup>

<sup>1</sup> Department of Computer Science and Engineering,  
Fudan University, Shanghai, China  
{wwqi, ayzhou}@fudan.edu.cn

<sup>2</sup> School of math and information Science Henan  
University, Kaifeng, China

**Abstract.** In this paper we propose a novel model for XML data management, which is called TROS (Text Relational Object Semantic), to overcome the shortcoming of directly storing XML data in relational databases. TROS consists of three layers, the top of which is object semantic layer, the middle relational data management layer and the bottom is text file management layer. We also consider the implementation issues of this model. By analyzing the semantics of queries posed by users, we propose a novel object-relational index structure, which can efficiently improve query performance by avoiding join operations between relations. In the model, relational queries result support corresponding document content's queries.

## 1 Introduction

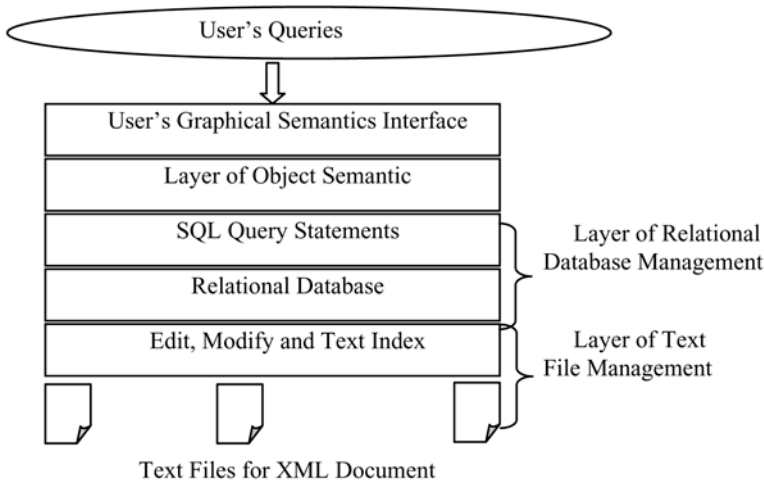
Flexibility of XML data structure caused by diversity of requirements in reality makes it quite difficult to store and manage XML data solely by relational databases<sup>[1-3, 5-8]</sup>. For example, in many XML documents, there is a large amount of text and other unstructured information that cannot be well normally formed in a relational database. Even for data of well-formed structure, to store it in a relational database, we have to split it into different tables to implement its hierarchical structure with key references between those tables. In fact, many efforts have been put into store and query XML data totally by relational ways.

There are many cases when XML data exists in the form of documents, which results in that it just too inconvenient for users to retrieve and management.

In this paper, we propose a novel model for managing XML data called TROS (stands for Text Relation Object Semantic). Within this model, we can manipulate XML data as it were in text files; meanwhile relational techniques are also adopted. Users can edit and search the text part of an XML document as they do on usual text files; and the data-centric part of the document is stored and managed in relational databases. By combing the two management methods, we can summarize the semantic and syntactic information of XML documents and propose the concept of Object Semantic Tree to provide supports for users to query the XML document.

---

\* This work is supported by the National Natural Science Foundation of China under Grant No. 60228006, and the National Hi-tech R&D Program of China under Grant No. 2002AA116020.



**Fig. 1.** Framework of the TROS model

## 2 The TROS Model

### The Basic content

1. Based on the schema and statistical information of an XML document, we propose the structure of object semantic layer, which is a synopsis of the content and structure of the XML document as DataGuides in [4]. The structure is described as a tree of object semantic, each node of which corresponds to a class of objects in the XML document.
2. With the XML document unchanged, we encode each object in the XML document, which corresponds to a class in the object semantic tree. The objects are encoded increasingly according to their position in the document.
3. We generate the relational schema from the object tree by defining a table for each node in the object tree. To maintain the relation between the content in tables and those in the XML documents, we store the content's position in XML document into the relational tables.
4. A novel object-relational index structure is defined in such model.
5. As for the queries under such model and index structure, those posed on the data-centric content can be answered within relational databases; Join operations between tables are not necessary in query evaluation any more.

## 3 The Object Semantic Layer

### 3.1 Element Tree

First, we parse the schema of XML documents to build corresponding element tree. In element tree, each node corresponds to an element described in the schema document, edge denote hierarchy of the elements. There are some nodes that have no concrete

element content and attribute value, only to indicate structure of the element tree. We call them empty nodes. In figure 2, the nodes filled with grey are empty nodes. To uniquely identify each element in an element tree, we number each element node as shown in Figure 2 additional a code for every node. We number nodes of the same level from left to right increasingly one, and add a separator to code of different level node, then every element node in an element tree is uniquely identified by its code. According to the code of every element, we can rapidly get the common deepest ancestor node for any element nodes in an element tree by computing their the longest common prefix. This code is simple and general [9], but its application is key. In our element tree, every element node includes the element name, type, code, occurrence (1,\*,+,?), empty or solid, all attribute name and type that can be get by parsing XML schema. Here we assume occurrence of all attribute is one. We denote by  $E$  an element tree.  $d$  denote the sum of all nodes in the element tree  $E$ . Any node  $e_i \in E$ ,  $E = (e_1, e_2, e_3, \dots, e_i, \dots, e_d)$ ,  $1 \leq i \leq d$ , where  $i$  is the sequence number of every node by pre-order traversing the element tree  $E$ .

### 3.2 Automatic Match Tree

We get the content of every element node only from the XML schema is insufficient in order to choose a better storage and index mechanism.

To get more information of every element node which occur in corresponding XML documents, we must scan original XML documents. For example occurrence sum of every element in the XML document, which element is document-centric and what is their size range etc. Due to semi-structure and self-describing characteristic of XML data, it is difficult to get the statistic data efficiently. In TROS model, we use different processing method to management document-centric and data-centric data. To obtain more details about the data types described in an XML document, we have to scan the original document. Because of complexity of the schema, variety of data types and large data amount of XML documents, it is necessary for us to employ efficient approach to collect information from the documents. Here, we present a structure of match tree and its match algorithm. The algorithm requires sequence scan corresponding XML documents one pass to get all statistic information what user need.

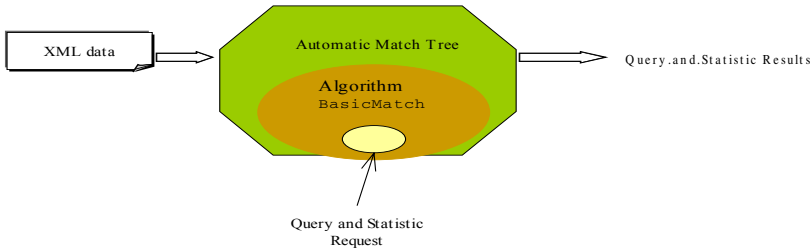
Traversing an element tree in pre-order can determine the emergency order of tags in the corresponding XML document. The uncertainty of the tags, i.e., those marked with '\*', '?' or '+', are key issues for us to make statistic, query and manage the XML documents. We consider that appearance sequence of tag in a XML document is according to depth-first search sequence of corresponding element tree, but indeterminate condition appear when occurrence of the element node is indeterminate (\*, +, ?). Because we don't predicate occurrence situation of the corresponding tag whose occurrence is indeterminate in XML document, so we need adjust match order according to the tag that is read in the document.

By taking the XML document as input and its element tree as a receiver, we can carry out matching analysis according to depth-first search sequence of corresponding element tree. Under such a condition, we can draw the following conclusions:

If the node of element tree matches with corresponding tag in the XML document, we continue matching according to the sequence of pre-order traversal to the element tree. Under such a condition, we can draw the following conclusions:

1. For an node of uncertainty in an element tree, if it is not found corresponding matching tag in XML documents, then the next matching position stays it's the nearest sibling node. If it has no sibling node, then the next matching position stays its nearest ancestor node marked with '\*' or '+'. If there is no such node, then the next matching position is null.
2. For leaf node of uncertainty marked with '\*' or '+' in an element tree, if matching then its next matching position is the self. The next matching position for other nodes is according to the sequence of pre-order traversal to the element tree.

Based on the above analysis, we can attach next matching position information to corresponding node in an element tree. The next matching clue is depicted as solid lines in Fig.3. To store the next match position of corresponding node in an element tree, we add a **nextmatch** field to each node. We call the additional next match information element tree **Automatic-match tree**.



**Fig. 2.** Application of the automatic-match tree

With **Automatic-match tree**, the matching process can be as follow, For each tag in the given XML document

1. If the tag is not matched with current node in the element tree, and the next matching position of current node is not null, we can get the next matching position and continue with the matching process. If the next matching position is null, then we have to find the next matching position by traversing the element tree by pre-order, and continue with the matching process.
2. If the leaf node that is being matched in the element tree is marked with '\*' or '+', then the next matching position is the leaf node itself, else if the next matching position of the currently matched leaf node is not null, then we should fetch the next matching position of this node. Otherwise, the next matching position of currently matched node is the next position of the node according to pre-order search sequence.

Because Automatic-match tree we can get statistic information for our special application goal by scanning the XML document only **one pass**. Here, it is an efficient tool by it we can get corresponding statistic information and creating corresponding index structure.



### 3.3 Trimming Element Tree

To trim an element tree of the XML documents, we inline those nodes whose occurrence is just one into their parent node, and retain those whose occurrences are not fixed, i.e. marked with ‘\*’, ‘+’ or ‘?’ .Fig.3 depicts an example of object semantic tree. Here, we present the trimming approach is not the same as that given in [3]. The nodes whose occurrence is optional (as marked with ‘?’) will not be embedded in our approach. We will get object semantic tree by applying such an approach on the element tree of XML documents. As for implementation, we add `inlinelev` and `inlinepointer` field to each node in the element tree. Its `Inlinelev` field stores the sum of its embedded up level and the `inlinepointer` field stores the pointer that point to its child node that will be embedded. Adding inline information to the element tree depicted in Figure 3 where the dashed lines with arrow indicate the inline clues, and the bold and italic numbers in the node indicate the sum of its embedded up levels. By the inline information, we can integrate the element tree into an object semantic tree. We call the procedure that transform an element tree  $E$  into an object semantic tree  $O$  *integration* which is expressed  $Inte(E) \Rightarrow O$  . By *integration* any element node  $e_i$  ( $1 \leq i \leq d$ ) is included in a special object node  $o_j$  ( $1 \leq j \leq f$ ) ,  $Inte(e_i) \rightarrow o_j, i \geq j, o_j \subseteq E$  ,where if  $o_j = (e_j, \dots, e_p)$   $e_j$  is the first element node which belongs to the object  $o_j$  .

We denote by  $O$  an object semantic tree,  $f$  denote sum of all nodes of the object semantic tree  $O, O = \{o_1, o_2, \dots, o_j, \dots, o_f\}, f \leq d, o_j$  is a node of  $O, o_j \in O$  , called a kind of object class , where  $j$  is sequence number of every object by pre-order.

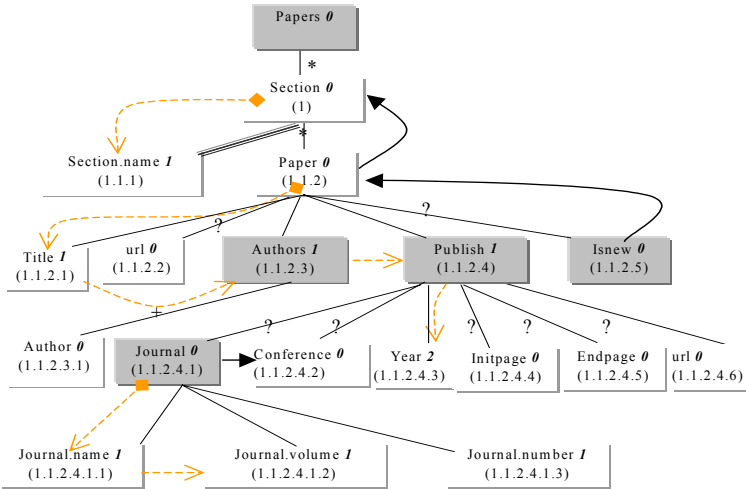


Fig. 3. A simple element tree

## 4 View of Object Semantic Tree

From the view of the object semantic tree, we can consider the content of an XML document as the entities set of all object classes described in the object semantic tree.

Each node in the object semantic tree is corresponding to a kind of object described in the XML document. For example, an abstract object semantic tree shown in Figure 4 there are seven kinds of object,  $A, B, C, D, E, F$  and  $G$ . We consider the XML documents as a set of the object entities.

We encode each entity for each kind of object class in the XML document. These object entities in the same object class in the XML document are encoded increasingly according to their position in the XML documents. Figure 4 illustrates an example of the XML documents and its object semantic tree. Any object  $o_j \in O$ ,  $Sum(o_j)$  denote the sum all entities of  $o_j$  in corresponding documents,  $o_j(en_i)$  denote a entity  $i$  of  $o_j, 1 \leq i \leq Sum(o_j)$ , where  $i$  is the code of the entity.

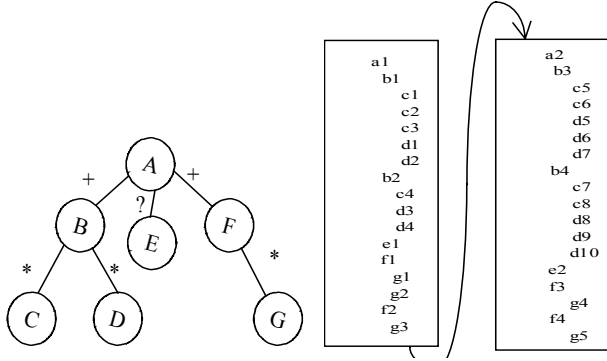


Fig. 4. An XML document example and its object semantic tree

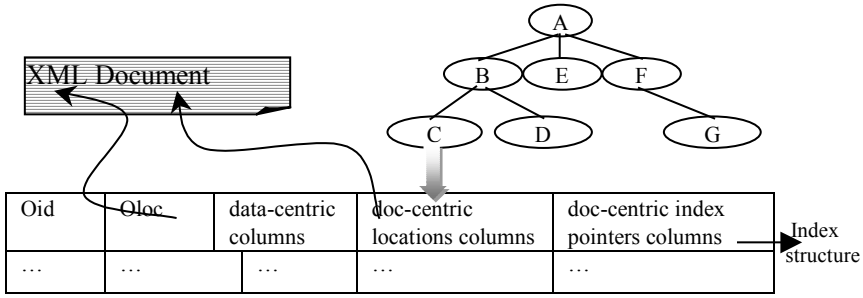
## 5 Generating Relational Schema

We generate the corresponding relational schema according to every object class in the object semantic tree. Each  $o_j \in O$  corresponds to a relational schema expressed as  $rs_j$ ,  $o_j \rightarrow rs_j$  is a one-to-one map,  $RS$  is the set of all relational schema generated by all object classes in a given object semantic tree  $O$ .  $OR$  denote the one to one object-relation map, it denote the map relation between  $O$  and  $RS$ .  $OR(o_j) \rightarrow rs_j$ ,  $OR(O) = RS$ . Because  $o_j \subseteq E$ ,  $o_j = \{e_j \dots e_p\}$ . Here, if  $e_q$  ( $j \leq q \leq p$ ) is data-centric element, its content in corresponding XML document is directly loaded into corresponding relation table. If  $e_q$  is not data-centric element (e.g. document-centric), its concrete content isn't loaded into relation table, for example its content is a book. We only store corresponding pointer information. By which we can rapidly find its concrete storage location and its indices. The relational schema definition is depicted in Figure5.

## 6 Queries of XML Data

### 6.1 Analysis of Queries

In the TROS model, user submitting a query called  $Q$ . Its restriction condition refer to element set called  $Re = \{\dots e_p, \dots, e_j, \dots, e_h, \dots\} \subseteq E$ ,  $Inte(Re) \Rightarrow R_o \subseteq O$ ,  $R_o$  denote the referring to object set of a query  $Q$ . Target of the query refer to the element set called



**Oid:** ID for every object entity of corresponding object class in object semantic tree

**Oloc:** Location of the object entity in XML document

**Data-centric columns:** Columns for data-centric elements, one column for each data-centric element in the corresponding object class.

**Doc-centric locations:** Doc-centric elements' location in XML document, one column for each the element in the corresponding object class, if they exist.

**Doc-centric index pointers:** Pointing to text Index structure of the corresponding doc-centric column.

**Fig. 5.** Object semantic tree and its relation schema

$T_e = \{\dots, e_h, \dots, e_b, \dots, e_l, \dots\} \subseteq E$ ,  $Inte(T_e) \Rightarrow T_o \subseteq O$ ,  $T_o$  denote the object set referred to by  $Q$ . We can express user submitting a query as  $Q(R_e, T_e)$  from view of element tree or  $Q(R_o, T_o)$  from view object semantic tree. Here,  $Cou(R_o)$  denote the object sum of the set  $R_o$ . Given a query  $Q(R_o, T_o)$ , if  $R_o = T_o$ ,  $Cou(R_o) = 1$ ,  $R_o = \{o_j\}$ ,  $OR(\{o_j\}) \rightarrow \{rs_j\}$  then we only query a relation table  $rs_j$  to get query result. This is the optimal case, because we don't need join table. Unfortunately, considering the characteristics of XML data, this would be a quite special case. More often, we have to evaluate the queries by joining lots of tables.

If  $Cou(R_o \cup T_o) \neq 1$ , then  $Cou(OR(R_o \cup T_o)) \neq 1$ , the query referring to the objects are more than one object, the usual approach is to join tables by columns defined for storing the hierarchical information, though relational schema is also generated by inline[3] operations, which can avoid many potential fragments. This will reduce the join operations, which will improve the query performance. But the general approach is that using joining operation of the relational tables express path expression of XML queries. It expresses hierarchical structure of XML data with flat relational tables.

In TROS model, by the semantic logic analysis of users queries, we know that the key problem is that how to generate logic connection between the target object class and the restriction object of user's query. We can draw a conclusion that the common nearest ancestor node is the key. In a object semantic tree  $O$ , given a query  $Q(T_o, R_o)$ , we can get the common nearest ancestor node  $w$  of the set  $T_o \cup R_o$ . We can describe a user's query completely as  $Q(T_o, R_o) \rightarrow w$  in a given object semantic tree  $O$ , where  $T_o \subseteq O$ ,  $R_o \subseteq O$ ,  $w \in O$ . Following discussion, we presume given an object semantic tree  $O$ .

## 6.2 Object-Relation Index

By an object semantic tree  $O$  and its corresponding relation set  $OR(O) \Rightarrow RS$ , given a query  $Q(T_o \cup R_o) \rightarrow w$ , if  $Cou(R_o \cup T_o) \neq 1$ ,  $Cou(OR(R_o \cup T_o)) \neq 1$ , how to build their

connection for the different relations. By above analysis, we build object-relation index for every inner node in object semantic tree. The corresponding object-relational index of the XML document example in Figure 4 as shown in Figure 6. In object-relation index table of an inner node, the column sum is the sum of its descendant nodes. Each row of the table corresponds a node in the object semantic tree, the values of which is the maximum code of its descendent nodes.

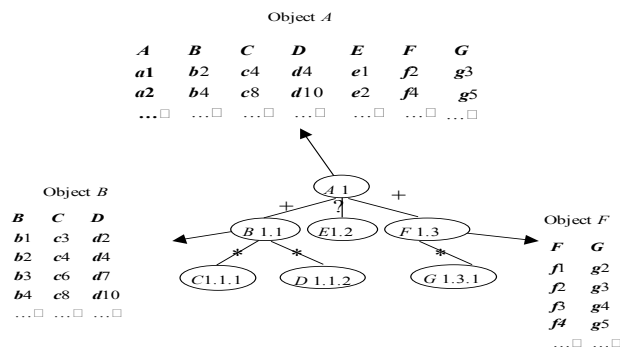


Fig. 6. Object-relation index tables of an object semantic tree

### 6.3 Queries with Object-Relation Index

With object-relation index, we can find in table corresponding to the restriction object class the code of those objects that satisfy the conditions. No matter what the target object class and restriction object class would be, we are always able to find the their common ancestor node  $w$ . In the indexing table corresponding to these common ancestor nodes, we can find those rows containing code of each class of restriction object. The codes of the target objects in such rows are ones of those objects satisfying the restrictions. We can find the objects' details in their corresponding tables. No join operations are performed in such a evaluation process.

For example, consider a query based on object semantic tree in Fig. 6, If the target object is  $C$ , the restriction object is  $G$ , and the nearest common ancestor of  $C$  and  $G$  is  $A$ .

To evaluate such a query, we first find in the table corresponding to  $G$  the code of objects satisfying the condition, to say,  $\{g1, g3\}$ . Then we find in object-relation index table corresponding to  $A$  the rows containing  $g1$  or  $g3$ , e.g.,  $a1$ . Then we can find in the row  $a1$  the code of  $C$ , to say,  $c4$ . Finally we can get the detailed information of  $c4$  in the relational table corresponding to object class  $C$ .

If there are more than one restriction object classes in the query, e.g.,  $G$  and  $E$ , we should begin to search in the relational table corresponded to the restriction object class whose occurrences are the least in the document. For example, in figure 6 the object  $E$  occurs less than object  $G$ , then we should search in the relational table corresponded to object  $E$  to find codes of object entities that satisfy the restriction on object  $E$ . Based on those codes found, we then search in the index table corresponded to their common ancestor node  $A$  to find the range of object  $G$ 's code. Within this range, we lastly search in the relation table corresponded to object node  $G$  to find the range of code of object  $G$  satisfying the restriction given by the user.

For other restriction object classes, we can perform the above process repeatedly until we get the code's range of the last restriction object class in its corresponding relational table. In the object-relational index table corresponded to the nearest common ancestor node, we can lastly get the range of the target object's code from that of the last restriction object. With such an approach, we cut the search space drastically by not only avoiding joining relational tables, but also reusing the code ranges of last restriction object. If there is doc-centric information in the restriction object class, we push such classes to the tail of the processing sequence so that we only need search text in the narrowest scope. If an XML document is of deep hierarchy, i.e., its object semantic tree is of large dept, and includes a long amount uncertain information (\*, +, ?), a query may cover target object classes and restriction object classes that of great distance in the object semantic tree. If we employ relational techniques, we have to join many tables. Here by employing object semantic layer and object-relational index, we can avoid a large amount join operations. The query performance would not decrease along with growth of the query path. At the same time, we using the query result of data-centric Information stored in relational table support the query to document-centric information.

## 7 Conclusions and Future Work

This paper proposes a novel model for managing XML data called TROS that combines the text characteristics and matured relational techniques. We also propose the concept of object semantic layer to depict the general structure of XML data. Based on such a model, we propos a novel object-relational index structure to index uncertain information of the XML data. Thanks to such a model and index structure, it is unnecessary to carry out join operations to evaluate queries. By proposing object semantic layer and analyzing the semantic of user's queries, we can keep away from the concept of query path. Users can describe what they want only by the contents presented in the object semantic tree. It is not necessary for them to know information about positions or relations.

In the future we will:

1. Summary the uncertain information in an XML document according to the node types in object semantic trees. Based on such statistical information, we can generate more efficient relational schema.
2. Adjust the object-relational index according to queries, e.g., we can pick out frequently used index fragment to improve query performance.
3. Elaborate the evaluation of user query to optimize queries in depth.
4. Design the graphical semantic interface.

## References

1. A. Deutsch, M. Fernandez, and D. Suciu. Storing semistructured data with STORED. In *Proceedings of the Workshop on Query Processing for Semistructured Data and Non-Standard Data Formats*, pages 431–442, 1999.

2. D. Florescu and D. Kossmann. Storing and querying XML data using an RDBMS. In *Bulletin of the Technical Committee on Data Engineering*, pages 27–34, September 1999.
3. J. Shanmugasundaram, K. Tufte, C. Zhang, G. He, D. J. DeWitt, and J. F. Naughton. Relational databases for querying XML documents: Limitations and opportunities. In *The VLDB Journal*, pages 302–314, 1999.
4. R. Goldman, J. Widom, DataGuides: Enabling Query Formulation and Optimization in Semistructured Databases, R. Goldman, J. Widom. In *VLDB 1997*
5. A. Zhou, H. Lu, S. Zheng, Y. Liang, L. Zhang, W. Ji, Z. Tian: VXMLR: A Visual XML-Relational Database System. In *VLDB 2001*. 719-720
6. P. Bohannon, J. Freire, P. Roy, and J. Simeon. From XML-Schema to Relations: A Cost-Based Approach to XML Storage. In *ICDE, 2002*.
7. Y. Chen, S. B. Davidson, and Y. Zheng. Constraint Preserving XML Storage in Relations. In *WebDB, 2002*.
8. Y. Chen, S. Davidson, C. Hara, and Y. Zheng: RRXS: Redundancy reducing XML storage In relations In *VLDB, 2003*.
9. L. Guo, F. Shao, C. Botev, and J. Shanmugasundaram, XRANK: Ranked Keyword Search In *SIGMOD, 2003*.

# Optimized Query Translation Strategy for XML Stored in Relational Database\*

Hongzhi Wang, Jianzhong Li, and Zhenying He

P.O.Box 318, Harbin Institute of Technology, 150001, China  
{Wangzh,lijz}@hit.edu.cn, hzy\_hit\_cn@sina.com

**Abstract.** XML is an important standard of data exchange and representation. XML database also becomes important as web database. As a mature database system, using relational database to support XML data may bring some advantages. In this paper, based on a relational storage structure support XPath query effectively, an optimized query translation strategy is presented as well as optimization techniques. This method converts XQuery to variable forest. Optimization operations are applied on the variable forest. The optimization is based on the semantics of XML document such as schema of original XML document. The VF can be converted to SQL easily. This method translates XQuery to a set of SQL statements and generates final result through the result of them. Our method adapts to the XML database with relational storage supporting effective query process to XML documents with various schemas and size.

## 1 Introduction

XML is widely used as the format of data representation and exchange of web. The requirement of querying XML database has increased. XQuery[7] emerges as the standard XML query language. The language supports many features of querying XML data.

The storage strategies of XML includes three types, flat text, OO DBMS and relational DBMS [21]. As a type of mature DBMS, relational database has its special advantages such as efficiency query process, concurrency control and so on. Commercial relational database products are widely used by many users. Using them to support the storage and query of XML data will result in many advantages. In fact, some commercial relational DBMS has already presented mechanism to support XML such as DB2 [18], Oracle [19] and SQL Server. But the XML processing mechanisms embedded in existing relational DBMS only support part of the features of XQuery or XML document with special schema.

Our goal is to design a XML query wrapper on existing relational DBMS to support the storage and flexible query of XML document with various schemas and sizes. This paper focuses on the translation technology from XQuery to SQL. With an ef-

---

\* Supported by the National Natural Science Foundation of China under Grant No.60273082; the Defence Pre-Research Project of the 'Tenth Five-Year-Plan' of China no.41315.2.3

fective storage structure supporting XPath well, the translation technologies are designed. The translation support main features of XQuery and XPath. Although existing DBMS may have general query optimizer, an optimization step of translation is necessary because that DBMS do not know the semantics information of the XML document itself such as schema information. Our translation method has following advantages:

The translation method is independence to the relational DBMS and the result can be converted to the final result with only wrap but not any other computation. It accelerates various types of XPath queries by converting XPath query to range query. With semantics of XML document, the optimization of the translation is applied to decrease the number of join, nested queries and the size of medium result so that the execution efficiency of translation result is accelerated.

In addition, this paper offers the following contributions:

- The storage structure is further optimized to support XML documents with variable schemas and sizes.
- An entire translation process is presented to assure an XQuery query can be converted to an effective SQL.
- Some optimization strategies are designed. These strategies are not only for the optimization for the translation on our storage structure but also adaptive for that on various structures.

The organization of this paper is organized as follows. The storage structure and the architecture of query processor are presented in section 2. In section 3, translation and optimization method are described. A performance experiment is presented in section 4. Section 5 summaries related research work. Section 6 draws the conclusion.

## 2 Framework of the System

In this section, a framework of the system is introduced. The system is built on relational database and a storage strategy independent to the schema is used. Storage structure and query process steps are introduced in this section respectively.

### 2.1 Relational Storage of XML documents

The storage of the system follows the method presented in [17]. The basic idea is to store a XML document to three relations, tag relation ( $r_{tag}$ ), node relation ( $r_{node}$ ) and value relation ( $r_{value}$ ). In  $r_{node}$ , each node of XML tree corresponds one record with preorder rank and postorder rank of this node stored. With these two values, the query of step in XPath can be converted to range query on  $r_{node}$ . The preorder rank of parent of a node is stored to be used to distinguish parent-child and ancestor-descent relationship. The advantage of this structure is that it is easy to express various axes of XPath. And the expression of XPath focuses on relation  $r_{node}$ , which has only numerical values and is fast to execute join operation on. The basic structure of the three relations is shown in the following three tables.



**Table 1.** Relation of tag code ( $r_{tag}$ )

name	Meaning
tagName	The name of a tag
tagNo	The code tagName corresponds to

**Table 2.** Relation of nodes ( $r_{node}$ )

name	Meaning
docID	ID of XML document
tagNo	The code of the node
nodePre	The preorder rank of the node
nodePost	The postorder rank of the node
par	The preorder rank of the parent of the node
isAttr	Whether the node is

**Table 3.** Relation of value ( $r_{value}$ )

name	Meaning
docID	ID of XML document
nodePre	The preorder rank of the node
Value	The value rank of the node

**2.2 An Overview of Query Engine**

Basic idea of query process is to express query plan with relational operators. During the translation, the optimization is based on the semantics of XML document, such as the original schema and statistics information. The steps of query process are shown in fig. 1.

Before query translation, the XQuery expression should be parsed and preprocessed.

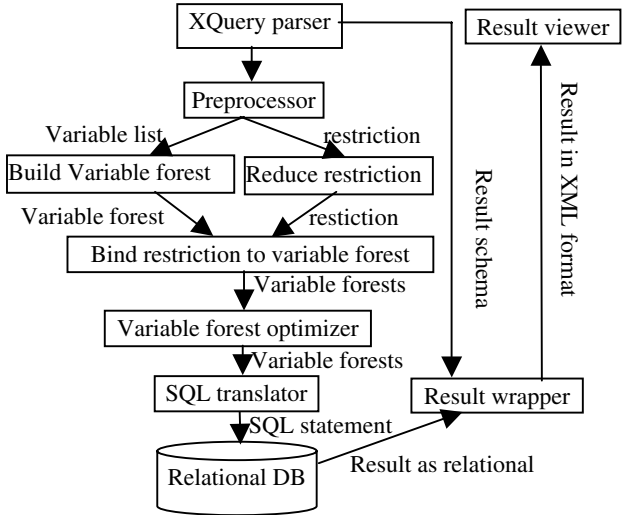
The first step of query translator is to extract the variables and their relationship from the query. They are represented as a tree, as is defined as variable tree. Each node represents a variable with its path in the query. More than one tree can be extracted from the query. The results of the trees are to be join in semantics.

All of variable trees of a query form a forest, as is defined as variable forest. This part will be described in section 3.1

And then the condition of the query is transformed into the form of DNF. Each clause of DNF is CNF. Each clause is to be generated on forest. For the optimization, some clause should be merged. The description of this part will be presented in 3.2.

The next step is to add each CNF clause of restriction clause to the forest. Additional nodes are added to the forest. Restrictions are also added to existing nodes.

With schema information, internal query plan is optimized in optimizer. Some optimize strategies are introduced in 3.5. And then optimized tree is translated to SQL



**Fig. 1.** Query Process steps

statement that can be executed in traditional relational database. The translation process is given in 3.4. The last step is to wrap the final result to the format of XML. The method of wrapper is presented in section 3.6.

### 3 Query Translation

Before query translation, the query should be preprocessed. Besides computing expression with only constant and reduce constant restriction, the code of tags exist in the XML query is directly obtain from  $r_{tag}$  so that it is not necessary to join  $r_{node}$  and  $r_{tag}$  during the query process. And the variables in group by, order by clauses are put in variable list of the parsed query.

#### 3.1 Construction of Variable Forest

The first step is to convert the query to variable forest containing several variable trees, which represents the relationship between the variables in the query.

**Definition 3.1** Variable Tree (VT for short): A Variable Tree of a query  $Q$   $T_Q=(r, V, E)$ , where  $r$  is the root of the tree,  $V$  is the set of nodes of the tree and  $E$  is the edge of the tree such that:

- Each  $v \in V$  can be represented as (ID, var, path, res, isresult, agg), where ID is the identify of the  $v$  which is uniform in the whole variable forest, var is the name of variable  $v$  binding to, path is the XPath expression of the var in  $Q$  and res is the restriction of this var. If  $v$  is a leaf node, res is the comparison of var and a constant. Otherwise res is the existing restriction of some child of  $v$  in  $T_Q$  or the relationship between the children of  $v$  in  $T_Q$ . property *isresult* represents if the variable exists in the final result. Property agg represents the aggregation function with id of group by nodes on this node.
- Each  $e \in E$  is labeled pc(for parent child relationship), ad(for ancestor descendant relationship), ad\* (for self-or descendant relationship) or the description of XPath.

□

**Definition 3.2** Variable Forest (VF for short): A set of variable trees  $F_Q=(S_T, R)$ , where  $S_T$  is the set of variable trees in  $F_Q$  and  $R$  is the set of relationships between the trees in  $S_T$  in  $F_Q$ .  $R$  is represented as the restriction to connect the VTs. □

The generation of a VF is to extract the variables from query and construct VTs. At first, for each independent variable that is obtained from for and let clause, a VT is built with ID, var and path expression. It is noted that the path expression is absolute path from root to the node var standing for. Then the trees are merged based on the path relationship stated in the query. When two VTs are merged, an edge connecting is added to  $E$  with its label representing the relationship between connected nodes. For construction of the result, nodes with virtual variables that are built from the path expression in result clause. And then the IDs of variables are filled back to restrict construction statement.

### 3.2 Reduction of Restrictions

During the process of parser, the restriction is converted to the conjunction normal form of disjunction normal form. The restriction can be represented in form of  $(r_{11} \wedge r_{12} \wedge \dots \wedge r_{1n1}) \vee (r_{21} \wedge r_{22} \wedge \dots \wedge r_{2n2}) \vee \dots \vee (r_{m1} \wedge r_{m2} \wedge \dots \wedge r_{mnm})$ . In order to optimize the query, the restrictions should be reduced.

**Definition 3.3** (Contain): Two restriction statement  $c_1$  and  $c_2$ , if the result of applied  $c_2$  on data must be a subset of applied  $c_1$  on the same data. It is defined that  $c_1$  contains  $c_2$ .  $\square$

**Definition 3.4** (Conflict): Two restriction statement  $c_1$  and  $c_2$ , if the result of applied  $c_2$  on data and that of applied  $c_1$  on the same data has no intersection. It is defined that  $c_1$  and  $c_2$  are conflict.  $\square$

The reduction of restrictions is to find the contain relationship between the DNFs. The algorithm of reduction process is shown in fig.5. The array valid to represented the state each CNF. Compare( $c_1$ ,  $c_2$ ) is the function that judges the relationships between two atom clause and return 1 if  $c_1$  contains  $c_2$ , return -1 if  $c_2$  contains  $c_1$ , return 2 if  $c_1$  and  $c_2$  are conflict, return 3 if  $c_1$  and  $c_2$  are equivalent, return 0 if there is no these two relationships between  $c_1$  and  $c_2$ .

### 3.3 Bind Restrictions to Variable Forest

Each disjunction normal form  $r_i = r_{i1} \wedge r_{i2} \wedge \dots \wedge r_{imi}$  of restriction is added to the VF. The restricted VF generated by  $r_i$  is  $VF_i$ . Internal plan  $p_i$  is translated from each  $VF_i$  respectively. Final result is the union of the result of all the internal plans.

The process of adding  $r_i$  to VF is to apply each  $r_{ij}$  to VF. The application is processed according to the variables in each  $r_{ij}$ .

- If  $r_{ij}$  contains just one variable or XPath expression that has been bound to some node  $n$  in VF, the restriction  $r_{ij}$  is add to res property of  $n$ .
- If  $r_{ij}$  contains variables or XPath expressions that have not been bound to any node  $n$  in VF, a node  $n$  binding to a virtual variable is added to corresponding VT in VF. In this instance, if  $r_{ij}$  has only one virtual variable, the restriction is added to res property of  $n$  directly.
- If  $r_{ij}$  contains more than one variable or XPath expression, all variables or XPath expressions are bound to the VF at first. And then, each variable  $v$  is check to determine which VT it belongs to. If all the variables belong to the same VT  $T_k$ , restriction  $r_{ij}$  is added to the nearest ancestor of them in  $T_k$ . Otherwise, a connection node is built to connect all the VTs that  $r_{ij}$  has variable binding to.

### 3.4 Translation to SQL Statements

The unit of SQL statement process is the ID of a node, nodePre is used to represent the id of a node. The translation course of a  $PT_i$  corresponding to  $VF_i$  is to traverse  $PT_i$  depth-first. Based on  $PT_i$ , in each variable bounded node, a SQL statement to represent the access the node is generated according to the generating method in  $PT_i$ . For each multiple-variable restriction bound node, a SQL statement to represent the connection of the variables. The connection is by nested query. The result of the translation is a set of SQL statements. Each of them has a uniform SID.

All the SQL statements form a partial ordering relation  $<$ . If the result of statement with  $SID_i$  is the table of statement with  $SID_j$ ,  $SID_i < SID_j$ . All the SIDs are topological sorted to assure all the statements have their table during process. Based on the partial ordering relation, all the queries can be compounded into one that is in form of nested query. It is noted that some relational DBMS have restriction on the length of query. In this instance, temporal table has to be used.

#### 3.4.1 Conversion of XPath Expression

The translation of one XPath Expression is presented in this section. Each step of the expression corresponds to a query. Result of the former is considered as the table of the next step. The query is in form of nested query. The restriction translated from axis of a step is shown in table 4, as is called  $c_a$ . The restriction translated from position restriction is shown in table 5, as is called  $c_p$ . And the restriction translated from value restriction is shown in table 6, as is called  $c_s$ . If there are more one  $c_{si}$  or  $c_{pi}$  of one node  $i$ , in order to keep the correct of result, they should be processed one by one with nested query. The restriction of the  $i$ th step of XPath Expression  $p$  is  $C_i = c_{ni} \wedge c_{ai} \wedge c_{fi} \wedge c_d$  in which  $c_{fi}$  is the first restriction of  $c_{si}$  or  $c_{pi}$  in this step,  $c_d = "r1.docID=r2.docID"$  and  $c_{ni}$  is the restriction for the name of the tag of this step. The translation result of this step is  $q_i$ . The translation steps can be expressed as following process. In this description, it is suppose there is just one  $c_{si}$  or  $c_{pi}$  of each step.

**Table 4.** Conversion of Axis Step

a/Child::b	b.nodePre>a.nodePre and b.nodePost<a.nodePost and b.par=a.nodePre
a/descendant::b	b.nodePre>a.nodePre and b.nodePost<a.nodePost
a/Parent::b	b.nodePre=a.par and b.nodePost>a.nodePost
a/Ancessor::b	b.nodePre<a.nodePre and b.nodePost>a.nodePost
a/following::b	b.nodePre>a.nodePre and b.nodePost>a.nodePost
a/preceding::b	b.nodePre<a.nodePre and b.nodePost<a.nodePost
a/Following-sibling::b	b.nodePre>a.nodePre and b.nodePost>a.nodePost and b.par=a.par
a/preceding-sibling::b	b.nodePre<a.nodePre and b.nodePost<a.nodePost and b.par=a.par

**Table 5.** Conversion of Position Restriction

[position()=c]	(select count(t. nodePre) from $q_a$ as a, $q_f$ as t where t.nodePre>a.nodePre and t.nodePost>a.nodePost and t.par=a.par and a.docID=t.docID)=c-1.
[position()=last()]	Not exist (select nodePre from $q_a$ as a, $q_f$ as t where b.nodePre>a.nodePre and b.nodePost>a.nodePost and b.par=a.par and a.docID=t.docID)

**Table 6.** Conversion of Value Restriction

[b]	Exist(select t.nodePre from $q_a$ as a, $q_b$ as t, $r_{tag}$ as v where t.nodePre>a.nodePre and t.nodePost<a.nodePost and t.par=a.nodePre and a.docID=t.docID)
[b='c']	Exist(select t.nodePre from $q_a$ as a, $q_b$ as t, $r_{value}$ as s where t.nodePre>a.nodePre and t.nodePost<a.nodePost and t.par=a.nodePre and and t.nodePre=s.nodePre and s.value='c' and a.docID=t.docID)

a and b in translated restriction in table 4 is the alais of the query to obtain node a and b. The b in table6 can be an XQuery expression. The  $q_a$  in translated restriction in table5 an table 6 represents the SQL expression to process the last restriction. The  $q_f$  in table6 is the SQL expression of the last step. The  $q_b$  in translated restriction in table 6 is the SQL expression to obtain b.

### 3.4.2 Translation of Internal Node and Multi-node Restrictions

The restriction in non-leaf node of VF is multi-node restrictions. The translation process of them is presented here.

The process of translating a internal node is shown in fig.6. For translated a internal node n, at first, the translated SQL statement of each of n' children  $c_i$  in VT is joined with rnode to obtain the nodes binding to n and in the path from root to nodes in  $c_i$ , the result of the join is  $s_i$ . The “from” objects of the SQL statement in internal node of VF are the result of the join of  $s_i$  with the restriction the conjunction of n.res and the expression representing that the record in all  $s_i$  have the same node binding to n.

In order to avoid confusion, during the process, an alias with corresponding id in VF is given to selected nodePres representing selected nodes.

On the process of the top level restriction, the projection operator is applied to only selected nodes whose isresult property are True with all the SQL statement of all the VTs as the tables and the top level restriction as the restriction of the final query.

### 3.5 Optimization of Internal Plan

It is noted that the SQL statement translated by the method in 4.4 from internal plan with the method stated in 4.3 may result in too many join operators and nested query. In this section, some optimization strategies are applied to accelerate the query process.

**Shorten of the Path:** As is stated in 4.4, the XPath query will result in nested query or join operation. It is noted that with schema information, the path can be shortened.[9]

**Definition 3.5** (full/part cover) For an XPath expression segment  $p$  with its start point  $st$  and the end point  $as$  is defined as a set  $S$ , if  $p$  can match all the paths from  $st$  to each node  $n$  in  $S$  in the schema. The segment  $p$  is called a full covered segment. Otherwise,  $p$  is called a part covered segment.  $\square$

The shorten process is to check the set of paths satisfying XPath expression in schema. If all the path in schema satisfies XPath expression, it only needs a selection operation  $r_{node}$ . Based on this idea, the XPath expression can be shortened. It is supposed that there is no restriction on any step of the XPath expression. In this algorithm, the XPath expression of each node is divided into several segments. The full cover and part cover property to the schema of these segments is interphase. Each full covered segment is processed by a selection operation directly. Each part covered segment is converted to the nested query by the method in 4.4. This strategy is effective for long path query.

**Reduce the Number of Joins:** It is known that in relational DBMS, nested query is often converted to join [16]. The join operation may be the operation with the most cost. Because the internal node has been already selected in the result, the id of the nodes are kept during the latter translation steps.

**Reduce the Size of Intermediate Result:** The intermediate results will be joined. The reduction of the size of intermediate result will accelerate the query process. The reduction is implementation by change the nested query order. If some child  $n'$  of  $n$  in a VT has restriction on,  $n$  is obtained from  $n'$  and the other children of  $n$  is obtain from this set of  $n$ . Thus, the size of intermediate result of  $n$  is decreased.

## 4 Experiments

### 4.1 Experimental Environment

The experiments are performed on a PC with AMD 1G CPU and 256M main memory. Operation system is Microsoft windows XP Professional and develop platform is Visual C++ 6.0. The backend database is Microsoft SQL Server 2000. The database is stored in local computer. Our problem connects the database with ODBC.

**Data Set:** XMark[17] is a famous benchmark of XML data management. It covers many features of XQuery. We use XMark as our data set with factor 1.0. The size of original XML document is 111M.

**Queries:** All the 20 queries presented in XMark are used for experiments.

**Database Setup:** Index is built in nodePre, nodePost and tagNo column of  $r_{node}$  and ragNo columns of  $r_{value}$ .

### 4.2 Performance Result and Analysis

The performance result is shown in table7.

**Table 7.** Performance result

Query	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10
Time(s)	2.995	38.576	3.565	197.534	3.685	0.03	0.12	82.689	597.645	666.726
Query	Q11	Q12	Q13	Q14	Q15	Q16	Q17	Q18	Q19	Q20
Time(s)	179.128	135.985	1.822	75.499	9.073	1.993	0.37	0.931	1.292	0.591

With query analyzer of SQL server, the internal query plan of each query can be viewed. From the result, the efficiency of the query executing in real environment on commercial database is acceptance. Q4 is slower is because the estimate result of the intermediate result is different to the real instance. The processes of Q8-Q11 are slower because the join of XML trees results in too many joins with some very large intermediate result ( for an example, during the process of Q9, the size of temp table have reached 2G!). Q14 is slower for the comparison between strings without text index on value property of  $r_{value}$ .

### 5 Related Work

There are many storage structures supporting XML documents. [1] and [2] considers large XML document and store XML document in the relations that are partitioned based on the schema of XML document. [6] and [10] is an extension of this method to self-tuning of the storage structure. These methods are not adaptive to the instance of storing many XML documents with various schemas. Query process in these storage structure results in many join operations. It is hard to translation XQuery to SQL for various instance of schema. More adaptive storage structure and query translation is presented in [3] and [4]. The method is to store each edge of the XML document. The shortcoming is that the query process efficiency on this storage is low. [5, 11] use interval encoding to represent the ancestor and descendance. XRel [5] is to encode the path and store the code and the extent of each node. But the translation detail of XQuery is not presented in [5]. The research work of [12] and [13] focus on the ordering of the nodes of XML document stored in relational database.

Another research work related to this paper is publishing XML view on relational database. [14] and [15] is work in this field. Some of the idea of query translation can be referred. But the essential difference is special storage structure can not be built to accelerate XML query process.

There are also many research of optimization strategy of XML query process. Some work is independence to the storage such as [9, 22]. They can be improved to support optimization of query process on XML stored in relational database.

## 6 Conclusions

XML is an important standard of data exchange and representation. XML database also becomes important as web database. As a mature database system, using relational database to support XML data may bring some advantages. In this paper, based on a relational storage structure support XPath query effectively, an optimized query translation strategy is presented. Optimization strategies based on semantics of XML document are presented. Our method adapts to the XML database with relational storage supporting effective query process to XML documents with various schemas and sizes. The translation method is independent to DBMS. With various connections, our system supports the query to various databases.

## References

1. A. Deutsch, M. F. Fernandez, D. Suciu, Storing Semi-structured Data with STORED, SIGMOD Conference 1999.
2. J. Shanmugasundaram, K. Tufte, C. Zhang, G. He, D. J. DeWitt, J. F. Naughton, Relational Databases for Querying XML Documents: Limitations and Opportunities. VLDB 1999.
3. S. Abiteboul, P. Buneman, D. Suciu. Data on the Web: From Relations to Semistructured Data and XML. Morgan Kaufmann Publishers. 2000.
4. Ioana Manolescu, Daniela Florescu, Donald Kossmann. Answering XML queries on heterogeneous data sources. In Proc. of VLDB 2001.
5. M. Yashikawa et al. : XRel: A Path-Based Approach to Storage and Retrieval of XML Documents using Relational Databases. TOIS2001.
6. P. Bohhanon et al. From XML Schema to Relations: A Cost-Based Approach to XML Storage. In the Proc. of ICDE 2002.
7. World Wide Web consortium. XQuery 1.0: An XML Query Language.  
<http://www.w3.org/TR/xquery/>.
8. World Wide Web Consortium: XML Path Language (XPath) 2.0.  
<http://www.w3.org/TR/xpath20/>.
9. Guoren Wang, Mengchi Liu, Jeffrey Xu Yu, Bing Sun, Ge Yu, Jianhua Lv, Hongjun Lu. Effective Schema-Based XML Query Optimization Techniques. In Proc. of IDEAS 2003.
10. Zhengchuan Xu, Zhimao Guo, Shuigeng Zhou, Aoying Zhou. Dynamic Tuning of XML Storage Schema in VXMLR. In Proc. of IDEAS 2003.
11. David DeHaan, David Toman, Mariano P. Consens, M. Tamer Ozsu. A Comprehensive XQuery to SQL Translation using Dynamic Interval Encoding. In Proc. of SIGMOD 2003.
12. Surajit Chaudhuri, Raghav Kaushik, Jeffrey F. Naughton. On Relational Support for XML Publishing: Beyond Sorting and Tagging. In Proc. of SIGMOD 2003.
13. Igor Tatarinov, Stratis Viglas, Kevin Beyer, Jayavel Shanmugasundaram, Eugene Shekita, Chun Zhang. Storing and Querying Ordered XML Using a Relational Database System. In Proc. of SIGMOD 2002.
14. Jayavel Shanmugasundaram, Jerry Kiernan, Eugene Shekita, Catalina Fan, and John Funderburk. Querying XML views of relational data. In Proc. of VLDB2001.



15. Mary Fernández, Yana Kadiyska, Dan Suciu, Atsuyuki Morishima, WANG-CHIEW TAN. SilkRoute: A Framework for Publishing Relational Data in XML. *ACM Transaction on Database Systems* 27(4) 2002.
16. Cliff Leung, Hamid Pirahesh, Praveen Seshadri, Joseph Hellerstein. Query Rewrite Optimization Rules in IBM DB2 Universal Database. *Readings in Database Systems*, 3rd ed. Morgan Kaufmann Publishers. 1998.
17. Torsten Grust. Accelerating XPath Location Steps. In *Proc of SIGMOD 2002*.
18. S. Banerjee, V. Krishnamurthy, M. Krishnaprasad, and R. Murthy. Oracle8i ?The XML Enabled Data Management System. In *Proc. of ICDE'00*. 2000.
19. J. Cheng, J. Xu. XML and DB2. In *Proc of ICDE'00*. 2000.
20. Albrecht Schmidt, Florian Waas, Martin Kersten, Micheal J. Carey, Ionana Manolescu, Ralph Busse. XMark: A Benchmark for XML Data Managemetn. In *Proc of the 28<sup>th</sup> VLDB conference*, 2002.
21. Feng Tian, David J. DeWitt, Jianjun Chen, Chun Zhang. The Design and Performance Evaluation of Alternative XML Storage Strategies. *SIGMOD Record* special issue on "Data Management Issues in E-commerce", March 2002.
22. J. McHugh and J. Widom. Query Optimization for XML. In *Proceedings of the Twenty-Fifth International Conference on Very Large Data Bases*, 1999.

# QReduction: Synopsizing XPath Query Set Efficiently under Resource Constraint

Jun Gao, Xiuli Ma, Dongqing Yang, Tengjiao Wang, and Shiwei Tang

The school of electronic engineering and computer science, Peking University  
Beijing 100871, China  
{gaojun,xlma,dqyang,tjwang}@db.pku.edu.cn, tsw@pku.edu.cn

**Abstract.** How to evaluate a massive XPath set over XML streams poses great challenges to database researchers. Current work chiefly focuses on evaluating efficiently massive XPath set to obtain precise results. The size of the input query set has a great impact on the resource requirement and the efficiency of evaluation. In this paper, we propose a novel method, QReduction, to obtain the synopsized XPath query set to represent the original query set, while at the same time to minimize the ‘precision loss’ caused by query set synopsis. QReduction discovers frequent patterns among the massive input XPath tree patterns first, and select query set synopsis from them based on a dynamic benefit model under resource constraints. Since frequent patterns discovery takes high complexity in QReduction, we propose optimization methods by pushing the constraints of QReduction into the discovery process. We propose 3 criteria, namely recall, precision and intersection to determine a better synopsis. The experimental results demonstrate that our method can produce a query set synopsis with high precision, recall and intersection under given resource constraints.

## 1 Introduction

As XML becomes a de facto standard for data representation and exchange on the Internet, many applications related with XML data stream have emerged, such as the content-based routing [4], document dissemination [4] etc. One of the key problems in such applications is how to evaluate a massive XPath set over XML data stream.

Current work focuses on how to evaluate the massive XPath set to obtain precise results for all queries. The size of the input query set has a great impact on the resource requirement and the efficiency at running time. However, in some applications, the precision of query result maybe not the most important factor. Taking subscribe/publish system as an example, users may tolerate imprecise results of a subscription to some extent. In other applications, the resource available is very limited. Those cases inspire us to synopsize query set with the minimized ‘precision loss’. In other words, we obtain another query set with a smaller size under given constraint to represent the original query set. The result of the original query, which can be calculated from the result of the result of query set synopsis, is as precise as possible.

## 1.1 Related Works

The related works include XML stream processing, tree aggregation, approximate query processing on data stream and XML similarity match.

Most of the currently proposed XPath evaluators over data stream are based on automata to obtain precise results. Some evaluators, such as XPush [3], are constructed on deterministic automata, which take exponential space in the size of input XPath set. Other evaluators, such as YFilter[4], are based on non-deterministic automata, hence the evaluation efficiency will decrease with the increase of input XPath set.

Tree patterns aggregation [6] shares the same goals as ours. Tree pattern aggregation depends on two basic operations, query containment and query minimization to aggregate XPath tree patterns. In our method, query containment can only be done on the sub tree and tree, hence leads to less complexity than the former work. In addition, the tree pattern aggregation has not taken the query distribution into account.

Data reduction techniques, such as wavelet [9], sampling [9], histogram [9], etc, are proposed to obtain the synopsized data for the original data set. However, these methods developed for relational data can not be applied on XML stream directly. In addition, data reduction methods cannot guarantee that all important data can be processed correctly.

Similarity query obtains some XML documents that do not meet the query requirement exactly. The methods used in similarity query, such as query rewriting, data relaxation, are as summarized in [1]. However, the work in [1] only deals with a single query.

QReduction takes frequent patterns as the basis for query set synopsis, while frequent patterns discovery is well studied in [2]. However, frequent pattern discovery is just the first step in QReduction, and QReduction improves the efficiency of discovery by applying the constraints of the whole QReduction process.

## 1.2 Contributions

This paper proposes a method, QReduction, to synopsise a given massive XPath {/, \*, [], } set under resource constraints. Specifically, the contributions are as follows:

QReduction provides a technique to evaluate massive XPath query efficiently on a very limited resources over XML data stream.

QReduction takes frequent patterns among the input XPath tree patterns as the basis for query set synopsis. The constraints on QReduction can be used to accelerate the frequent patterns discovery significantly.

QReduction takes a dynamic benefit model to select the synopsized query from the frequent patterns so as to improve the recall, precision and intersection of query set synopsis (defined in section 2).

The experiments in this paper show that QReduction can generate smaller size query collections and minimize the ‘precision loss’ caused by query set synopsis.

The rest of the paper is organized as follows: Section 2 presents QReduction to synopsise the input XPath set; Section 3 discusses some optimization strategies ap-

plied in QReduction. A performance study in section 4 demonstrates the validation of our methods. Section 5 concludes the paper and discusses the future work.

## 2 Query Set Synopsized from the Massive XPath Set and Evaluated over Data Stream

XML can be modeled as a node-labeled directed tree. XPath [5], the basic mechanism of tree navigation proposed by W3C, supports a number of powerful features. The former work just deals with some commonly used features [2, 3]. The features of XPath supported in this paper includes ‘/’ for parent/child relationship, ‘//’ for ancestor/descendant relationship, ‘\*’ for wildcards, and ‘[]’ for a branching condition.

The operator  $\{[]\}$  introduces much complexity in XPath processing. XPath without  $\{[]\}$  can be expressed by traditional automata, while XPath with  $\{[]\}$  can not. In the following, we call XPath with  $\{[]\}$  as *complex XPath*, and XPath without  $\{[]\}$  as *simple XPath*.

A query set synopsis is another query set with a smaller size and the result of the query set synopsis has some relations with that of the original query. Different methods produce different query set synopsis. In order to determine the better query set synopsis, we make the following criteria:

**Definition 1.** the recall, precision and intersection of query set synopsis  $P_2$  for query set  $P_1$

Given a data stream  $L$ , a query set  $P_1$ , query set synopsis  $P_2$  for  $P_1$ , mapping  $M$  takes the form  $\{q_1 \rightarrow q_2\}$ , where  $q_1 \in P_1, q_2 \in P_2$ . Suppose that subset  $S_1$  of  $P_1$  returns TRUE on  $L$ ; and subset  $S_2$  of  $P_1$ , calculated from the result of  $P_2$  and mapping  $M$  from  $P_1$  to  $P_2$ , returns TRUE on  $L$ . The recall of query set synopsis  $P_2$  for  $P_1$  is defined as  $|S_1 \cap S_2|/|S_1|$ , denoted as recall  $(P_2, P_1)$ . The precision of query set synopsis  $P_2$  for  $P_1$  is defined as  $|S_1 \cap S_2|/|S_2|$ , denoted as precision  $(P_2, P_1)$ . For each query  $p_i$  in  $P_1$ , we denote  $r_1$  as the result of  $p_i$ , and  $r_2$  as the result of  $p_i$  from the result of  $P_2$  and mapping  $M$ . We define  $S_3$  including query  $q_1$  with  $r_1=r_2$ .  $|S_3|/|P_1|$  is called the intersection between the two query sets, denoted as intersection  $(P_2, P_1)$ .

The problem can be stated as follows: Given a query set  $P_1$  and space constraint  $C$ , how to generate query set synopsis  $P_2$ , with the space cost of evaluator constructed from  $P_2$  meeting the space constraint  $C$  while maximizing recall  $(P_2, P_1)$ , precision  $(P_2, P_1)$  and intersection  $(P_2, P_1)$ .

We briefly summarize the method in this paper. Each XPath defined in our paper can be represented by tree pattern [7]. Sub tree  $s_1$  of tree pattern  $t_1$  can be used to represent  $t_1$ . Since there are many candidates sub trees to be used to represent  $t_1$ , we hope  $s_1$  can represent other tree patterns. In other words, sub tree  $s_1$  indicates a frequent pattern among the given XPath tree patterns. Hence the first step in our method is to find frequent patterns. The discovered frequent pattern can be used to be basis for query set synopsis, which is selected based on a dynamic benefit model based on the recall, precision and intersection of query set synopsis.

## 2.1 Discovery of Frequent Patterns among Tree Patterns

A Frequent pattern  $fp$  is a sub tree in at least one original tree pattern. In the following, we propose an Apriori-like method to discover frequent patterns among the input XPath tree patterns. First, we give some related notations.

Given an XPath tree pattern  $T$ ,  $T$  can be decomposed into a path set  $S: \{t_1, \dots, t_n\}$ , where the initial node of each  $t_i$  ( $1 \leq i \leq n$ ) is the root node of  $T$  and the end node of  $t_i$  ( $1 \leq i \leq n$ ) is the leaf node of  $T$ . Since there is no a branch node in decomposed path, the XPath in  $S$  is a simple XPath. The size of  $S$  is called the *weight* of tree pattern  $T$ . If XPath  $p_1$  is *contained* in XPath  $p_2$ , it means the results set of  $p_1$  is a subset of that of  $p_2$ .

**Definition 2 (*N*-order frequent pattern).** *N*-order frequent pattern takes the form  $\{c\} \rightarrow \{c_1, \dots, c_n\}$ , where  $c, c_1, \dots, c_n$  are complex XPath,  $c_i$  ( $1 \leq i \leq n$ ) is contained in  $c$ , the weight of  $c$  is  $N$ .

**Algorithm 1.** Discover ( $N, CQ, S$ ) discover *N*-order frequent patterns from XPath query set  $CQ$  with minimal support  $S$

Input: XPath query set  $CQ$ , support  $S$ , order  $N$

Output: *N*-order frequent patterns

- [1] Decompose each query  $p$  in  $CQ$  into a simple XPath set  $SQ$ ;
- [2] As for a simple XPath  $s_1$  in  $SQ$ , construct a candidate pattern in the form of  $\{s_1\} \rightarrow \{c_1, \dots, c_n\}$ , which indicates  $s_1$  is decomposed from  $c_i$ ,  $1 \leq i \leq n$ . We call  $\{s_1\}$  the *head* of the form, and  $\{c_1, \dots, c_n\}$  the *body* of the form. The size of the body is denoted as the *weight* of the pattern.
- [3] Let  $FP$  be the set of patterns, the weight of each pattern in  $FP$  is greater than  $S$ ; Let  $TMP$  be  $FP$ ;
- [4] For  $i=1$  to  $N$ 
  - a)  $FP = FP \text{ join } TMP$ ; // combine the heads and bodies of the two patterns.
  - b) Eliminate the pattern  $p$  in  $FP$ , where the weight of  $p$  is less than  $S$  or the order of  $p$  does not equal  $i$ ;
- [5] Let patterns set  $RS$  be  $FP$ , each pattern  $p \in RS$  takes the form of  $\{s_1 \dots s_n\} \rightarrow \{c_1, \dots, c_m\}$ ;
- [6] Combine  $s_1, \dots, s_n$  into  $c$  for each pattern  $p \in RS$ , each pattern  $p \in RS$  in the form of  $\{c\} \rightarrow \{c_1, \dots, c_m\}$ .
- [7] Output the final  $RS$  as *N*-order frequent patterns
- [8] End

Notice the pattern  $\{c\} \rightarrow \{c_1, \dots, c_m\}$  generated in Step 6. Since all simple XPath queries,  $s_1, \dots, s_n$ , are decomposed from  $c_1, \dots, c_m$ , it is easy to know that  $c_1, \dots, c_m$  are contained in  $c$ . In this way, the high cost of query containment is avoided in the whole process.

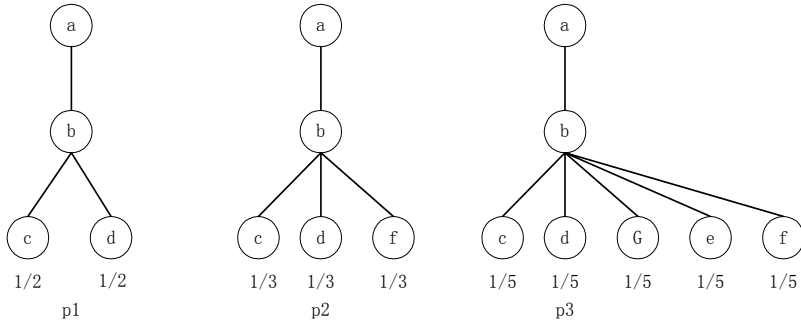
The above method just obtains *N*-order frequent patterns with the minimal support  $S$ . We need to run the discovery algorithm more than once to obtain frequent patterns with all orders.

## 2.2 Synthesize Query Set from Frequent Patterns under Resource Constrains

After the generation of frequent patterns with each pattern in the form of  $\{c\} \rightarrow \{c_1, \dots, c_m\}$ , the head of frequent pattern is used to be a candidate to be selected into query set synopsis.

**Definition 3.** (the similarity between sub-tree and tree). Given a tree  $T_1$  and a sub-tree  $T_2$  in  $T_1$ , the number of leaf node in  $T_1$  is  $N_1$ , the number of leaf node in  $T_2$  is  $N_2$ , the similarity between sub-tree and tree can be defined as  $|N_2|/|N_1|$ .

**Example 2.** given XPath  $p_1=/a/b/c[d]$ ,  $p_2=/a/b/c[d][f]$ ,  $p_3=/a/b/c[d][f][g][e]$ , we construct the following tree patterns:



**Fig. 1.** XPath tree patterns

XPath  $p_4='a/b/c[d]'$  is the head of a possible frequent pattern  $\{p_4\} \rightarrow \{p_1, p_2, p_3\}$  among three XPath tree patterns. However, the similarity between  $p_3$  and  $p_4$  is less than that between  $p_1$  and  $p_4$  or that between  $p_2$  and  $p_4$ . Such a case leads to different precision loss when  $p_4$  is used to represent them.

Notice that it is not necessary to define the similarity between two arbitrary XPaths, but that of sub-tree and tree in QReduction. The similarity, as one of the parameters of QReduction, is set by users to guide the system to adjust the precision of query set synopsis. Taking figure 1 as an example, if we assign 0.5 on the similarity threshold,  $p_3$  is removed from the body of pattern since the similarity between  $p_4$  and  $p_3$  is 0.4. In this way, we can control the 'precision loss' between the query set synopsis and the original query set.

After the removal of complex query which does not meet the similarity threshold from the body of frequent patterns, we need a criterion to decide which frequent pattern can be dealt first.

Benefit  $B$  of a frequent pattern  $fp$  can be calculated from the complex XPaths in the body of  $fp$ . Suppose that one complex query  $c_1$  in the body of  $fp$  can decompose into  $S$  with  $N$  simple XPaths, we assign the value of each part as  $(1/N)$ . The head of frequent pattern  $fp$  can decompose into  $F$  with  $M$  simple XPaths. If an equivalent mapping can be constructed from simple XPath in  $S$  to that in  $F$ , the benefit of frequent pattern  $fp$  to complex XPath  $c_1$  can be defined as the sum of values of mapped

simple XPath in  $S$  from  $M$ . Benefit  $B$  of frequent pattern  $fp$  can be denoted as the sum of benefit of  $fp$  to all complex XPath in the body. The cost  $C$  of frequent pattern  $fp$  can be denoted as the space the head query of  $fp$  occupies in the evaluator (including the states and transition rules in final evaluator). The benefit of frequent pattern  $fp$  per unit can be denoted as  $B/C$ .

Taking figure 1 as an example again, the benefit of  $p_4$  to  $p_3$  is  $2/5$ , the benefit of  $p_4$  to  $p_1$  is  $2/3$  and the benefit of  $p_4$  to  $p_1$  is 1. Hence, the benefit of  $p_1=1+2/3+2/5$ .

We select one frequent pattern with the maximal benefit of frequent pattern per unit. After one frequent pattern  $f_1$  is selected, frequent pattern  $f_2$  with the second highest benefit may represent the same complex XPath which can be represented by  $f_1$ . Such a case will lead to low recall of the generated query set synopsis when resources are limited. Hence, after one frequent pattern is selected, the benefit of the rest of frequent patterns should be calculated again. Thus, a dynamic benefit model is proposed in QReduction.

**Algorithm 2.** Query set synopsis generation

Input: an XPath query set  $CQ$ , a similarity threshold  $SIM$ , a space constraint  $SPC$ , a minimal support  $SPT$ .

Output: query set synopsis  $QS$

- [1] For (int  $N=1; N < M, N++$ ) //where  $M$  is maximal weight of query in  $CQ$ ;
  - a) Obtain all frequent patterns  $FP$  by discover( $N, CQ, SPT$ )
- [2] Eliminate complex XPath  $c_1$  in the body of frequent pattern  $fp \in FP$ , the similarity between the head of  $fp$  and  $c_1$  is less than  $SIM$ ;
- [3] Set the value of each simple XPath decomposed from complex XPath  $c_1 \in CQ$ ;
- [4] Sort frequent pattern  $fp \in FP$  based on the benefit of frequent pattern per unit;
- [5] Set  $C$  to be 0; //  $C$  is the occupied space
- [6] While  $C < SPC$ 
  - a) Add the frequent pattern with highest benefit to query set synopsis  $QS$ ;
  - b) Set the value of the related single XPath decomposed from complex XPath to be 0;
  - c) Recalculate the benefit of rest of frequent patterns and Resort them;
- [7] End

The benefit of each frequent pattern is calculated dynamically in step 6. That is, once a frequent pattern is selected, the value of related simple XPath decomposed from complex XPath is set to 0, and then the benefits of the rest of frequent patterns have to be recalculated to obtain the next frequent pattern with the highest benefit. Taking the case in figure 1 as an example, after  $p_4$  is selected, the value of all decomposed simple XPaths from  $p_1$  is set to 0, the value of the decomposed simple XPath ' $/a/b/c$ ' and simple XPath ' $/a/b/d$ ' in  $p_2$  is set to 0.

### 3 Optimization Method in QReduction

Algorithm 1 takes a high time complexity to discover frequent patterns among the input XPath tree patterns. Suppose that there are  $N$  candidate frequent patterns ini-

tially, the generation of 3-order frequent patterns takes  $O(N^3)$ , leading to a serious efficiency problem. In this section, we propose several strategies to improve the scalability of QReduction.

### 3.1 Optimization Method 1 in Frequent Patterns Discovery

Recall the process of QReduction as a whole. Suppose we want to find  $N$ -order frequent patterns with similarity  $SIM$ . The frequent patterns generated in algorithm 1 take the form  $\{s\}:-\{c_1, \dots, c_k\}$ . A complex XPath  $c \in \{c_1, \dots, c_k\}$  is removed from the body of the frequent pattern in algorithm 2 when the similarity between  $c$  and  $s$  is less than  $SIM$ .

Based on such an observation, we can push the similarity check at the stage of candidate patterns initialization so as to reduce the intermediate results in frequent patterns discovery.

**Definition 4.** ( $N$ -order candidate with similarity  $SIM$ ). Given order  $N$ , similarity  $SIM$ , a candidate frequent pattern  $fp$  in the form of  $\{s_1\} \rightarrow \{c_1, \dots, c_k\}$ ,  $N$ -order candidate with similarity  $SIM$  takes the form  $\{s_1\} \rightarrow \{d_1, \dots, d_p\}$ , where for any  $d \in \{d_1, \dots, d_p\}$ ,  $d \in \{c_1, \dots, c_k\}$ , the weight of  $d$  is between the  $N$  and  $(N/SIM)$ , .

**Theorem 1.** All  $N$ -order frequent patterns can be generated from the  $N$ -order candidates with similarity  $SIM$ .

**Proof.** Given an original candidate pattern with the form of  $\{s_1\} \rightarrow \{c_1, \dots, c_k\}$ , suppose the weight of  $c \in \{c_1, \dots, c_k\}$  is not between  $N$  and  $(N/SIM)$ , even if  $c$  appears in the body of one of the frequent patterns after algorithm 1,  $c$  will be eliminated in algorithm 2 because the weight of any validate complex XPath in the body of the pattern is between  $N$  and  $(N/SIM)$ , or else the complex XPath in the body does not meet the similarity requirement. Hence all  $N$ -order frequent patterns can be generated from the  $N$ -order candidates with similarity  $SIM$ .  $\square$

Theorem 1 sets a more rigid constraint on the initialization of candidate frequent patterns. In algorithm 1, a pattern can be a candidate pattern if it satisfies the minimal support requirement. If theorem 1 is applied, a pattern can be a candidate if the  $N$ -order candidate with similarity satisfies the minimal support requirement. In this way, the size of initial candidates and intermediate results are greatly reduced.

### 3.2 Optimization Method 2 in Frequent Patterns Discovery

The optimization method 1 can be used to reduce intermediate results significantly. It works better when order  $N$  is relatively small. However, as for the candidate pattern with a higher order, intermediate results can still be large enough. We need further work to reduce them.

Remind the goal of frequent patterns in our paper again. The frequent patterns can be used as basis for query set synopsis. When system resource is relatively limited, the recall of query set synopsis is one of the most important criteria. We expect to



avoid the overlapping of the complex XPath set in the body of different selected frequent patterns. Such a case inspires us to obtain frequent patterns in a reverse way. That is, we check whether a complex XPath can appear in one of the frequent patterns.

Formally, we reverse a candidate frequent pattern form  $\{s_1\} \rightarrow \{c_1, \dots, c_n\}$  into a pattern in a new form  $\{c_1\} \rightarrow \{s_1, \dots, s_m\}$ , where  $s_i$  ( $1 \leq i \leq m$ ) is a simple XPath decomposed from complex XPath  $c_1$ . As for a candidate pattern  $f_1$  in new form  $\{c\} \rightarrow \{s_1, \dots, s_m\}$  with size  $m$  of the body greater than  $N$ , we select  $n$  simple XPaths from the body of  $f_1$ , and check whether the selected XPaths can build a frequent pattern satisfying the support requirement. Once one valid frequent pattern is generated, it indicates that the complex XPath  $c$  will appear at least in one frequent pattern. Hence we stop checking and output the valid frequent pattern.

It will take the exponential time to obtain the optimal frequent pattern with the highest support. A suitable frequent pattern can be generated in polynomial time. In Qreduction, we make  $O(m)$  test for  $\{c\} \rightarrow \{s_1, \dots, s_m\}$  to find the frequent pattern  $f_p$  with the approximately highest support. Hence each complex XPath can generate one frequent pattern as basis for the generation of query set synopsis. As for a candidate pattern  $f_1$  in new form  $\{c\} \rightarrow \{s_1, \dots, s_m\}$  with size  $m$  of the body less than  $N$ , we discard  $f_1$  directly. We give a further proof in the following.

**Theorem 2.** Given order  $N$ , a candidate pattern  $f$  in the  $\{c\} \rightarrow \{s_1, \dots, s_m\}$ , where simple XPath set  $\{s_1, \dots, s_m\}$  can be decomposed from complex XPath  $c$ . If the size of  $\{s_1, \dots, s_m\}$  is less than  $N$ ,  $c$  will not exist in any valid  $N$ -order frequent patterns.

**Proof.** Suppose we obtain frequent patterns in a normal way by algorithm 2. Any  $N$ -order frequent pattern takes the following form  $\{s_{11}, \dots, s_{1n}\} \rightarrow \{c_{11}, \dots, c_{1k}\}$ , where the head of the frequent pattern contains  $n$  different simple XPaths, and any complex XPath in  $\{c_{11}, \dots, c_{1k}\}$  can decompose at least  $n$  simple XPaths. Hence, given a candidate pattern in a new form  $\{c\} \rightarrow \{s_1, \dots, s_m\}$ , if the size of  $\{s_1, \dots, s_m\}$  is less than  $n$ ,  $c$  will not appear in any  $N$ -order frequent pattern.  $\square$

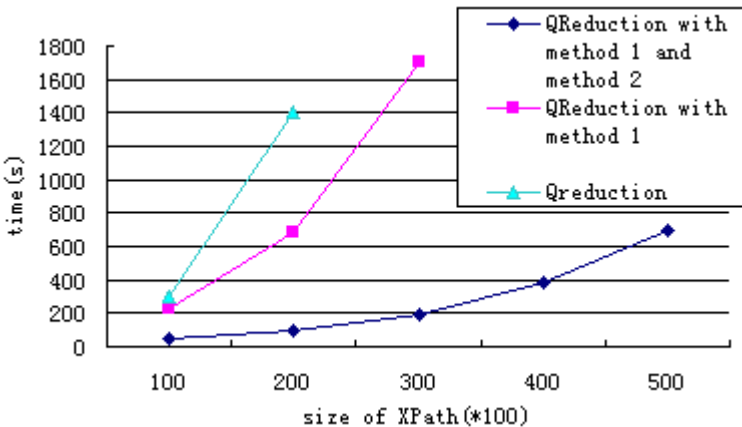


Fig. 2. Time cost in query set synopsisizing

## 4 Performance Study

QReduction is implemented by Java 1.31. Experiments were run on the Dell Optiplex GX260 with 512M of RAM and 2.0G Intel Processor running Windows 2000. We use XPath generator [8] to generate a massive XPath set.

We make experiments on the time complexity of query set synopsizing first. In the process, we focus on the effectiveness of optimization method 1 and method 2 in section 4. When the size of XPath set exceeds 20K, we stop QReduction without any optimization methods since it takes too much time. From the above results, method 1 and method 2 can improve the efficiency of frequent patterns discovery significantly.

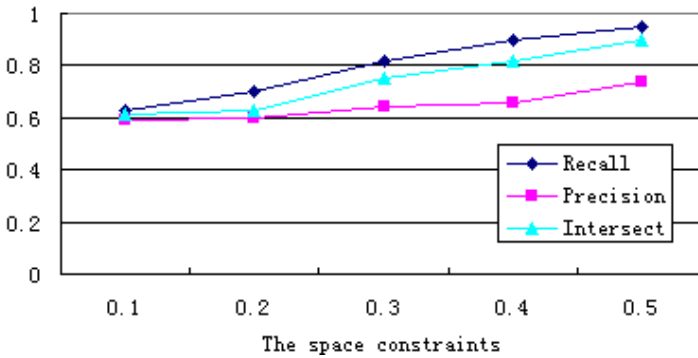


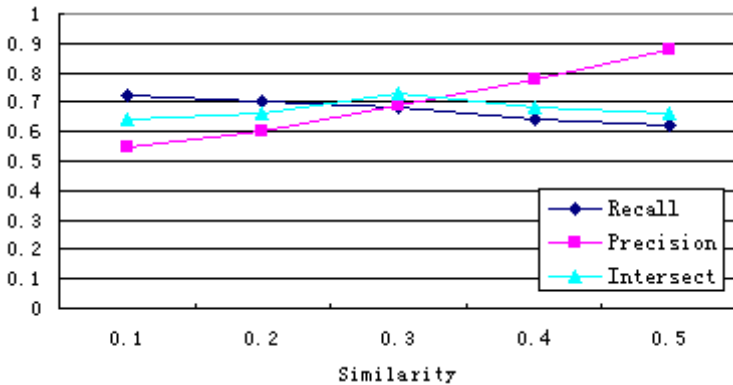
Fig. 3. Recall, precision and intersection changes with various space constraints

It can be observed that the recall, precision and intersection of query set synopsis change with various space constraints. In the above experiment, the space constraint is roughly represented by the ratio of the size of the query set synopsis to the size of original one. That is, when the space constraint is set to 10%, the size of query set synopsis is 10% of that of original query set. Such an assumption is different from our problem formula, but it is sufficient to demonstrate the effect of various space constraints in QReduction. In figure 4, the similarity is set to 0.2, the size of XPath set exceeds 20K, we notice that the recall, precision, and interaction increase when the space constraints increase.

We set space constraints as 20%, 20K XPath set, and obtain the recall, precision and intersect with the various similarities. From figure 5, the increase of similarity leads to the increase of precisions, but leads to the decrease of recall. Such result are the same as our assumption.

## 5 Conclusion

In this paper, we propose QReduction to generate query set synopsis from a massive XPath set so as to improve the efficiency and reduce the resource cost of evaluator. QReduction generates frequent pattern efficiently and select the synopsized query based on a dynamic benefit model from the frequent patterns. The final experiment results also demonstrate the validation of our method.



**Fig. 4.** recall, precision and intersection change with various similarities

## Acknowledgments

Authors were supported by supported by the National High Technology Development 863 Program of China under Grant No.2002AA4Z3440; the National Grand Fundamental Research 973 Program of China under Grant No. G1999032705

## References

1. Sihem Amer-Yahia, SungRan Cho, Divesh Srivastava: Tree Pattern Relaxation. In proceedings of 8<sup>th</sup> EDBT conference, 2002, pages 496-513.
2. Liang Huai Yang, Mong-Li Lee, Wynne Hsu: Efficient Mining of XML Query Patterns for Caching. In proceedings of 29<sup>th</sup> VLDB conference, 2003, pages 69-80.
3. Ashish Kumar Gupta, Dan Suciu: Stream Processing of XPath Queries with Predicates. In proceeding of SIGMOD conference, 2003, pages 419-430.
4. Yanlei Diao, Peter Fischer, Michael Franklin, and Raymond. YFilter: efficient and scalable filtering of XML documents. In proceeding of ICDE conference, 2002, pages 341-345.
5. J.Clark. XML Path language(XPath), 1999. available from the W3C, <http://www.w3.org/TR/XPath>.
6. Minos Garofalakis, Chee Yong Chan, WenFei fan, Pascal Felber, Rajeev Rastogi. Tree pattern aggregation for scalable XML data dissemination, In proceedings of 28<sup>th</sup> VLDB conference, 2002.
7. G.Miklau and D.suciu. Containment and equivalence for an XPath fragment. In proceedings of 21<sup>th</sup> PODS, 2002, pages 65-76.
8. C.Chan, P.Felber, M.Garofalakis and R.Rastogi. Efficient filtering of XML document with XPath expressions. In proceedings of 18<sup>th</sup> ICDE conference, 2002, pages 235-244.
9. Minos N. Garofalakis, Phillip B. Gibbon: Approximate Query Processing: Taming the TeraBytes. In proceedings of 27<sup>th</sup> VLDB Conference, 2001

# Extracting Key Value and Checking Structural Constraints for Validating XML Key Constraints\*

Yunfeng Liu, Dongqing Yang, Shiwei Tang, Tengjiao Wang, and Jun Gao

School of Electronics Engineering and Computer Science  
Peking University, Beijing, China  
yfliu@db.pku.edu.cn, tjwang@pku.edu.cn

**Abstract.** We propose an approach that can effectively validate key constraints over XML document by extracting key value and checking structural constraints. First we propose an XPath-based algorithm that can extract key values from the XML document and generate the corresponding key value document. Then we present how a key value document and its DTD can be designed to check whether predefined key constraints are satisfied. Last we draw an interesting conclusion that as long as XML keys can be expressed in XPath, the validation problem can be done by the XPath and the structural constraints checking.

## 1 Introduction

The role of XML in data exchange over the internet is evolving from one of merely conveying the structure of data to one that also conveys its semantics[1,2]. To address this problem, a number of semantic constraints specifications, especially for key definitions, have been proposed which include a series of key definitions for XML [3,4,5,6]. Recently many aspects of these proposals have been adopted within XML Schema [7], and the significance of XML key constraints has been recognized.

While key constraints validating problem has been well studied in the relational model, validating constraints over XML document appears to yield many new difficult problems to be faced. Although key definitions are being adopted by XML Schema, unlike structural constraints checking (DTD checking), there is yet no standard technique for validating that a document conforms to a set of XML keys[8].

In this paper, we propose an approach that can effectively validate key constraints over XML document by extracting key value and checking structural constraints. More specifically, we make the following contributions:

- We propose a XPath-based algorithm that can extracts key values from the XML document and generates the corresponding key value document.
- We present how a key value document and its DTD(We named KVD and KVD-DTD respectively) can be designed to check whether predefined key constraints are satisfied.

---

\* This work is supported by the NKBRSF of China (973) under grant No.G1999032705, the National '863' High-Tech Program of China under grant No. 2002AA4Z3440.

- We draw an interesting conclusion that a document  $D$  satisfies the predefined key constraints if and only if its KVD satisfies the KVD-DTD, which is a structural constraint checking problem. This conclusion reveals that as long as XML keys can be expressed in XPath the validation problem can be done by the XPath and the structural constraint checking.

We conduct a set of experiments and the experimental results generated from the real data reveal that the algorithms work well in practice.

**Related Work.** A number of definitions of XML keys and foreign keys have been proposed [3,4,5,6,7]. [9] studies the consistency of such specifications: given a DTD and a set of constraints, whether there exists a XML documents conforming to the DTD and satisfying the constraints. But[6] showed that for DTDs and arbitrary keys and foreign keys, the consistency problem is undecidable.

At present, several validation tools are being developed[8,10,11,12,13], but as[8] describes, most of them do not support regular expressions provided by XPath and the few have completely implemented XML Schema keys. Since the XML keys themselves are expressed in XPath and the structural constraint checking is a well solved problem, the validation using XPath based on DOM and the structural constraint checking is a principal motivation for our work.

## 2 XML Key Constraints and Validation

We begin with the brief definitions of the XML keys and the problem statement of key constraints validation which will be used throughout the paper.

**Definition 1. (XML Key)** An XML key can be written as:  $K: (C, (T, \{P_1, \dots, P_p\}))$

Where  $K$  is the name of the key, path expressions  $C$  and  $T$  are the context path and target path expressions respectively, and  $P_1, \dots, P_p$  are the key paths. A key is absolute if the context path  $C$  is the empty path  $\epsilon$ , and relative otherwise.

**Example 1.** Let us consider a sample document “companies.xml” whose tree representation is shown in Fig. 1 and we can define the keys for the document:

$KS_1: (/,(./company, \{./name\}))$        $KS_2: (/company,(./branch, \{./@ID\}))$   
 $KS_3: (/,(//Mgr, \{./name, ./office\}))$

The XML document describes the information about some companies. The meaning of the keys are as follows:

$KS_1$ : the key of the company is name;

$KS_2$ : in a company, the ID attribute of the branch can uniquely identify a branch;

$KS_3$ : the manager can be uniquely identified by the name and office within the whole document.

**Definition 2. (Key Satisfaction)** An XML tree  $\check{T}$  is said to satisfy a key  $K = (C, (T, \{P_1, \dots, P_p\}))$ , denoted  $\check{T} \models K$ , if and only if for each context node  $c$  and for any target nodes  $t_1, t_2$  reachable from  $c$  via  $T$ , whenever there is a non-empty intersection of values for each key path  $P_1, \dots, P_p$  from  $t_1, t_2$ , then  $t_1$  and  $t_2$  must be the same node.

**Definition 3. (Key Validation Problem)** The problem of key validation asks if given an XML tree  $T$  (generated from a document  $D$ ), and a set  $\Sigma$  of XML key constraints, whether the for every  $K \in \Sigma$ ,  $T$  satisfies  $K$ . We write  $D \models \Sigma$  if the implication holds for  $T$  satisfying all  $K$  in  $\Sigma$ .

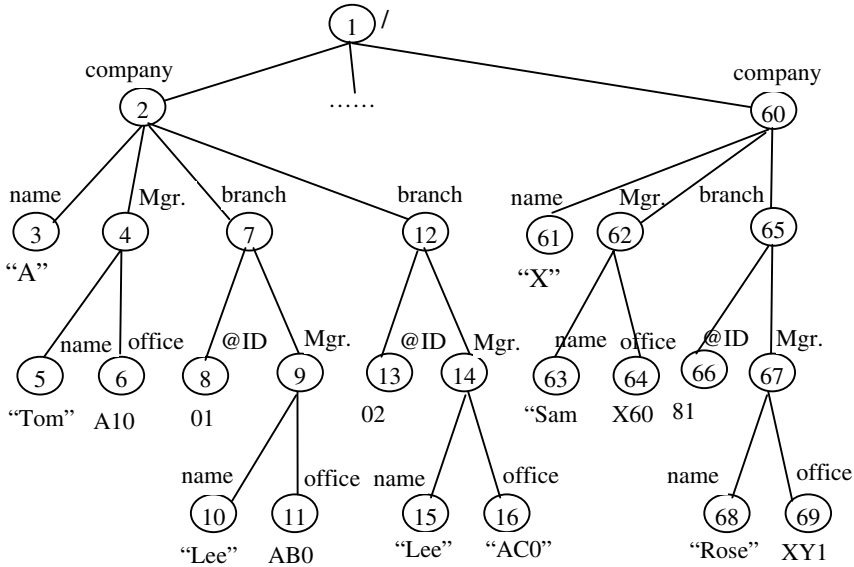


Fig. 1. Part of tree representation of an XML document

### 3 Validating Key Constraints by Extracting Key Value and Checking Structural Constraints

Although XML key definitions have been given, the question of how to build an efficient validator to best validate the constraints have not been solved. Few parsers can completely support XML Schema key. Key constraints validation problem needs to be well studied.

We focus on validating key constraints over XML document automatically. Since the XML keys themselves are expressed in XPath and the structural constraint checking is a well solved problem, the validation using XPath-based key value extraction and the structural constraint checking is a principal motivation for our work. In this paper, we present a new technique that can effectively validate key constraints over XML document by extracting key value and checking structural constraints.

#### 3.1 Overview of the Approach

The validation problem is parameterized by the XML document and the keys. The main steps are:

- First, the XML document will be parsed by the DOM Parser and a DOM tree will be generated as the input parameter for KVD Generator as well as XML keys.
- Second, using a XPath-based algorithm, the KVD Generator extracts key values from the DOM tree according to the definition of XML keys and generates the corresponding key value document KVD.
- Third, the Structure Validator checks whether the KVD satisfies the KVD-DTD using the structural constraint checking method.

3.2 Extracting Key Values from the XML Tree Using XPath

In this section, we propose a XPath-based algorithm that can extracts key values from the XML document and generates the corresponding key value document called KVD that can completely map the given XML key specifications.

**Example 2.** Return to our example 1, given a key specification  $KS_1: (/,./company, \{./name\})$ , the main part of the generated KVD document is written as Table 1.

Since the XML keys themselves are expressed in XPath and the structural constraint checking is a well solved problem, the validation using XPath based on DOM and the structural constraint checking is a principal motivation for our work. To extract key values from the XML tree effectively, we adopt the XPath-based primitives to do the job.

Table 1. Main part of the KVD document

< Key KID =1 >	
<ContextNode CID = 1>	
<KeyPath Number = 1 Value = “A”>	<TargetNode>2</TargetNode>
</ KeyPath >	
<KeyPath Number =1 Value = “X”>	<TargetNode>60</TargetNode>
</ KeyPath >	
</ContextNode>	
< /Key >	

Before presenting the Key-Value-Extraction Algorithm, we propose a set of XML Primitives based on XPath for operating xml document and generating the KVD.

**Definition 4. (XPath-Base Primitives)** We define the XPath-base Primitives which directly operate on xml document as follows:

- **Tagging** (D, path, tagName): add an element that its tag named tagName at position path to the XML document D.
- **AddAttribute** (D, path, attName, attValue): add a new attribute attName with attValue at position path to the XML document D.
- **GetNode** (D, path): get all the nodes under the path expression in the XML document D.

- **GetID** (D, node): get ID of the node in XML document D.
- **AddContent** (D, path, <tagName>val</tagName>): add the element whose name is tagName with content val under the path expression to the XML document D.

**Algorithm 1. (Key-Value-Extraction Algorithm)**

Table 2 is the Key-Value-Extraction Algorithm. The key idea of the algorithm is to extract key value information from the original XML document according to the predefined XML keys and compose a KVD using XPath-base primitives.

**Table 2.** Key-Value-Extraction Algorithm

<pre> <b>function</b> KeyValueExtraction (D,KeySet) /* Input: D, KeySet=K1,K2, ...Km (Ki = (Q,(Q',{P1, P2,...Pn}))) */ /* output: D' */ 1. Tagging (D', XMLKeys ); 2. For i=1 to m Do /*Begin to deal with Ki*/ 3. { 4. Tagging(D', /XMLKeys/, Key); 5. AddAttribute(D', /XMLKeys/ Key[i], KID, i); 6. ContextNode_Set = GetNode( D, /Q/ ); 7. For every ContextNode in ContextNode_Set Do 8. { 9. Tagging(D', /XMLKeys/ Key[i], ContextNode); 10. CurrentCID = GetID(D, ContextNode); 11. AddAttribute(D', /XMLKeys/ Key[i]/ContextNode, CID, CurrentCID); 12. For j=1 to n Do /* Begin to deal with Pj */ 13. { 14. Tagging(D', /XMLKeys/ Key[i]/ContextNode[CID=CurrentCID]/, KeyPath); 15. AddAttribute(D',/XMLKeys/ Key[i]/ContextNode[CID=CurrentCID]/KeyPath[j], Number,j); 16. Pj_Value_Set = GetContent(D, //ContextNode//Pj); 17. For every Pj_v in Pj_Value_Set do 18. { 19. AddAttribute(D',/XMLKeys/Key[i]/ContextNode[CID=CurrentCID]/ Key- Path[j],Value, Pj_v); 20. Tagging(D', /XMLKeys/ Key[i]/ContextNode[CID=CurrentCID]/, KeyPath); 21. AddAttribute(D',/XMLKeys/Key[i]/ContextNode[CID=CurrentCID]/KeyPath[j], Number,j); 22. } 23. TagetNode_Set = GetNode(D, //Context/Q'[Pj/text() = Pj_Value]); 24. For every TagetNode in TagetNode_Set Do 25. AddContent(D', /XMLKeys/Key[i]/ContextNode[CID = CurrentCID]/ KeyPath[j], &lt;TagetNode&gt;GetID(D, TargetNode)&lt;/TagetNode&gt;); </pre>
----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------



26.	} /*end of all the key path*/
27.	} /*end of all the context node*/
28.	TargetNode_ShareKeyValue = GetContent(D', //KeyPath[1]/TargetNode);
29.	For k = 2 to m Do
30.	TargetNode_ShareKeyValue = TargetNode_ShareKeyValue $\cap$ GetContent(D', //KeyPath[k]/TargetNode);
31.	if  TargetNode_ShareKeyValue  $\geq$ 1
32.	{ /*add all the key paths and the corresponding key values that generate the non- null intersection*/
33.	Tagging(D', /XMLKeys/KeyPath[1]/, TargetNode_ShareKeyValue);
34.	for every node in TargetNode_ShareKeyValue do
35.	{ Tagging(D', /XMLKeys/KeyPath[1]/TargetNode_ShareKeyValue, inter section);
36.	AddContent(D', /XMLKeys/KeyPath[i]/TargetNode_Intersection/, < T_node >GetID(D, TargetNode)</ T_node >))
37.	}
38.	} /*end of the TargetNode_ShareKeyValue*/
39.	} /*end of the Ki*/
40.	return(D');

**Table 3.** The main part of KVD document

< XMLKeys > < Key KID =1 > <ContextNode CID = 1> <KeyPath Number = 1 Value = “A”> <TargetNode>2</TargetNode> </ KeyPath > < KeyPath Number =1 Value = “X”> <TargetNode>60</TargetNode> </ KeyPath > </ContextNode> </Key > < Key KID =2 > <ContextNode CID = 2> <KeyPath Number = 1 Value = “01”> <TargetNode>7</TargetNode> </ KeyPath >	< KeyPath Number =1 Value = “02”> <TargetNode>12</TargetNode> </ KeyPath > </ContextNode> <ContextNode CID = 60> <KeyPath Number = 1 Value = “81”> <TargetNode>65</TargetNode> </ KeyPath > </ContextNode> </Key > < Key KID =3> ..... </Key > </XMLKeys >
----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Lines 2-39 in the algorithm deal with every XML key from  $K_1$  to  $K_m$ . Lines 6-27 begin to deal with every context node in XML key, finding all the target nodes identified by the different key value of key path and add the corresponding information to the KVD. Lines 28-38 compute the content of the TargetNode\_ShareKeyValue element and add lines 31-38 add all the key paths and the corresponding key values that generate the non-null intersection.

**Example 3.** Now let us return to our example again. Using companies.xml and the predefined keys in example 2 as parameters, according to KVD generating function, the main part of the KVD is in Table 3.

### 3.3 Checking Structural Constraint

In this section, we present how a key value document KVD and its DTD (We named KVD-DTD) can be designed to check whether predefined key constraints are satisfied. We define the KVD-DTD as Table 4 shows:

**Table 4.** KVD-DTD: The DTD for Key Value Document

```
<!DOCTYPE XMLKeys[
  <!ELEMENT Key(ContextNode+, TargetNode_ShareKeyValue*)>
  <!ATTLIST Key KID CDATA #REQUIRED>
  <!ELEMENT ContextNode(KeyPath*)>
  <!ATTLIST ContextNode CID CDATA #REQUIRED>
  <!ELEMENT KeyPath(TargetNode+)>
  <!ATTLIST KeyPath
    Number CDATA #REQUIRED
    Value CDATA #REQUIRED >
  <!ELEMENT TargetNode(#PCDATA)>
  <!ELEMENT TargetNode_ShareKeyValue (Intersection*)>
  <!ELEMENT Intersection(k_Path+, T_Node)>
  <!ELEMENT T_Node(#PCDATA)>
  <!ATTLIST k_Path
    Number CDATA #REQUIRED
    Value CDATA #REQUIRED >]>
```

**Table 5.** KVD-Structure-Validation Algorithm

```
function KVD-Structure-Validate (D',N)
/* Input: D', N:Number of XML Keys */
/* output:true or false */
1. for i = 1 to N
2.{
3. Targetnode_Set=/XMLKeys/KeyPath[i]/TargetNode_ShareKeyValue/ Intersection/
   T_node /;
4. if |Targetnode_Set| > 1 then
5.   return false;
6. }
7. return true;
```

**Algorithm 2. (KVD-Structure-Validation Algorithm)**

The structure validation is the last step of the whole process. The KVD-Structure-Validation Algorithm is shown in Table 5. The function returns true if the number of the child of element Intersection in KVD for each key is not greater than 1, or else the function returns false. In this way, we implement the key constraints validation through structural constraint checking.

**Example 4.** The validate function determines whether the Targetnode element in TargetNode\_Intersection element occurs only once. Continue the example 3, the validate function returns true.

**Example 5.** Imaging the value of the text node 16 in Fig. 1 is “AB0”. For the  $KS_3$ , there are two nodes 9 and 14 share the same key values for all the key path, so there is a Intersection element of Target\_ShareKeyValue in the KVD which does not conform to the KVD-DTD. We draw a conclusion that the companies.xml is invalid considering the predefined keys.

## 4 Analysis of Algorithm and Experimental Results

The following lemma and theorem prove the soundness of the algorithms.

**Lemma 1. (Soundness of the Algorithms).** A XML document D satisfies the predefined set  $\Sigma$  of key constraints if and only if its KVD (generated by the Key-Value-Extraction algorithm) satisfies the KVD-DTD(Checking by KVD-Structure-Validation algorithm).

Proof. First, we consider the “if” portion of the proof:

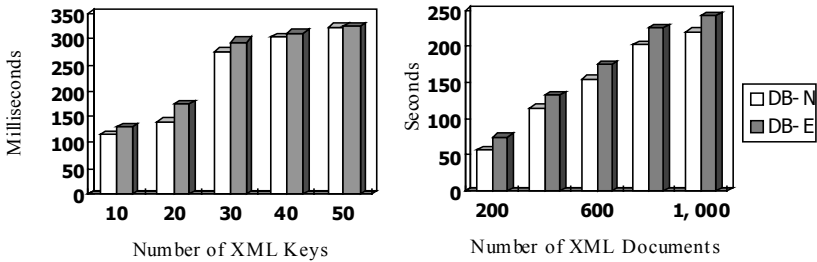
If a XML document D satisfies the predefined set  $\Sigma$  of key constraints, we assume that the its KVD does not satisfies the KVD-DTD. That is to say, the KVD-Structure-Validation algorithm will return false. According to the step 4 of the algorithm, we know that the  $|Targetnode\_Set| > 1$ , and there must be two T\_nodes below some KeyPath according to the step 3. Let us return to the Key-Value-Extraction algorithm, from step 31-38 we can say there must exists a target t within a context node c of an XML key k, the t shares some key value with another target node under c for every key path. That is to say, the XML document does not satisfy the K. But this is in contradiction to the condition of D satisfying the set  $\Sigma$  of key constraints. So the “if” portion is correct. So the “if” portion is correct and the “only if” portion is same as the above. These two portions prove soundness of the two algorithms.

Now we can state our main result for problem of validation key constraints over XML document in theorem 1.

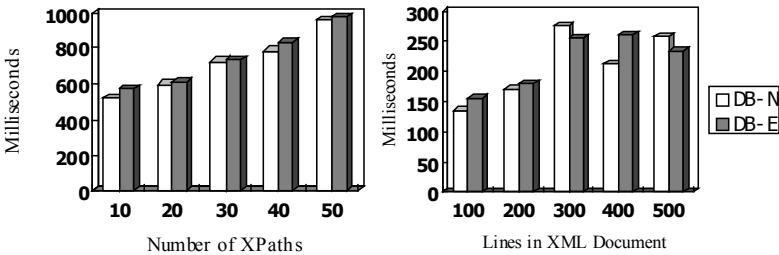
**Theorem 1.** As long as XML keys can be expressed in XPath, the XML key validation problem can be done by the XPath and the structural constraint checking.

Recently, we are developing CoXML, a system towards building an XML-based middleware system. We evaluate the performance of our method against using two real XML data sets (one for e-business DB-E and the other for news DB-N).

- We check the effectiveness of Key-Value-Extraction algorithm according to the numbers of key constraints and the numbers of XML documents. Fig. 2 (a) depicts the execution time of the Key-Value-Extraction algorithm with number of XML keys varying from 10 to 50. Fig. 2 (b) depicts the execution time for 10 XML keys with number of XML documents varying from 200 to 1000.
- We want to study the impact of the number of XPath queries on the performance of Key-Value-Extraction algorithm, and how the number of lines in XML document influences the performance of the algorithm. Here we choose the number of keys = 5, again based on the two database. Fig. 3 (a) depicts the execution time of the Key-Value-Extraction algorithm with number of XPath's varying from 10 to 50 and Fig. 3 (b) depicts the execution time with number of lines varying from 10 to 50.



**Fig. 2.** (a) Effect with number of keys (b) Effect with number of documents



**Fig. 3.** (a) Effect with number of XPath's (b) Effect with number of lines

The set of results conducted on the real XML data reveal that the two algorithms can be done in polynomial-time and works well in practice.

## 5 Conclusion

In this paper, we present a new technique that can effectively validate key constraints over XML document by extracting key value and checking structural constraints. First we propose a XPath-based algorithm that can extract key values from the XML document and generates the corresponding key value document. Then we present

how a key value document and its DTD can be designed to check whether predefined key constraints are satisfied. Last we draw an interesting conclusion that as long as XML keys can be expressed in XPath, the validation problem can be done by the XPath and the structural constraints checking.

## References

1. Michael Benedikt, Chee Yong Chan, Wenfei Fan, Juliana Freire, and Rajeev Rastogi. Capturing both Types and Constraints in Data Integration. In Proceedings of the ACM SIGMOD International Conference on Management of data. San Diego, California, USA, 2003
2. S. Davidson, W. Fan, C. Hara, and J. Qin. Propagating XML Constraints to Relations. In the Proceedings of the 19th International Conference on Data Engineering (ICDE'03)
3. W. Fan and J. Siméon. "Integrity Constraints for XML". In Proceedings of the 2000 ACM Symposium on Principles of Database Systems (PODS), pages 23-34., Dallas, TX, May, 2000
4. P. Buneman, W. Fan, J. Simeon, and S. Weinstein. Constraints for Semi-structured Data and XML. SIGMOD Record, 30(1), 2001
5. P. Buneman, S. Davidson, W. Fan, C. Hara, and W. Tan. Keys for XML. In Proceedings of the 2001 Proceedings of the tenth International Conference on World Wide Web, pages 201 - 210, Hong Kong, China, May 2001
6. P. Buneman, S. Davidson, W. Fan, C. Hara, and W. Tan. Reasoning about Keys for XML. In Proceedings of the 8th International Workshop on Data Bases and Programming Languages (DBPL'01), Rome, Italy, 2001
7. H. Thompson, D. Beech, M. Maloney, and N. Mendelsohn. XMLSchema Part 1: Structures. W3C Working Draft, April 2000. <http://www.w3.org/TR/xmlschema-1>
8. Yi Chen, Susan B. Davidson, Yifeng Zheng: XKvalidator: A Constraint Validator for XML. In Proceedings of ACM Conference on Information and Knowledge (CIKM 2002): 446-452
9. Marcelo Arenas, Wenfei Fan and Leonid Libkin. On Verifying Consistency of XML Specifications. In Proceedings of the Twenty-first ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS 2002), pages 259-270, 2002
10. Microsoft XML Parser 4.0(MSXML). <http://msdn.microsoft.com>
11. XML Schema Validator. <http://www.ltg.ed.ac.uk/ht/xsv-status.html>
12. XML Schema Quality Checker, February 2002. <http://www.alphaworks.ibm.com/tech/xmlsqc>
13. Béatrice Bouchou, Mírian Halfeld Ferrari Alves and Martin A. Musicante Tree Automata to Verify Key Constraints. In Proceedings of Sixth International Workshop on the Web and Databases (WebDB'2003). Pages 184-190. June, San Diego, CA, USA, 2003

# Estimating the Selectivity of XML Path Expression with Predicates by Histograms<sup>\*</sup>

Yu Wang<sup>1</sup>, Haixun Wang<sup>2</sup>, Xiaofeng Meng<sup>1</sup>, and Shan Wang<sup>1</sup>

<sup>1</sup> Information School, Renmin University of China, Beijing 100872, PRC  
{wangyu6, xfmeng, swang}@mail.ruc.edu.cn

<sup>2</sup> IBM Thomas J. Watson Research Center Hawthorne, NY 10532  
haixun@us.ibm.com

**Abstract.** Selectivity estimation of path expressions in querying XML data plays an important role in query optimization. A path expression may contain multiple branches with predicates, each of which having its impact on the selectivity of the entire query. In this paper, we propose a novel method based on 2-dimensional value histograms to estimate the selectivity of path expressions embedded with predicates. The value histograms capture the correlation between the structures and the values in the XML data. We define a set of operations on the value histograms as well as on the traditional histograms that capture nodes positional distribution. We then construct a cost tree based on such operations. The selectivity of any node (or branch) in a path expression can be estimated by executing the cost tree. Compared with previous methods (which ignore value distribution) our method offers much better estimation accuracy.

## 1 Introduction

As XML becomes the standard for data exchanging over the Internet, much research has been devoted to the efficient support of XML queries. We study XML query optimization based on selectivity estimation. If the branch with less selectivity is queried first, the intermediate result will have a relatively small size and the query efficiency can be improved. XML data is a hybrid of structures and values, some of which are highly correlated. Disregarding values' hierarchical distribution may affect the accuracy of selectivity estimation. For instance, assume we have an XML document as that of Fig. 1(a), and we want to estimate the selectivity of path expression `item[type='appliance'] [price>500]`. For this particular example, the selectivity of 'appliance' on item is 50% (2 of the 4 items are of type `appliance`), and that of 'price>500' is also 50%, because there is a high correlation between item `price` and item `type`. Since no

---

<sup>\*</sup> This research was partially supported by the grants from 863 High Technology Foundation of China under grant number 2002AA116030, the Natural Science Foundation of China (NSFC) under grant number 60073014, 60273018, the Key Project of Chinese Ministry of Education (No.03044) and the Excellent Young Teachers Program of M0EPRC (EYTP).

value distribution information is available, the best we can assume is that the values are independent. Thus, the selectivity of the path expression on item is estimated as  $50\% \times 50\% = 25\%$ , while the true selectivity is 50%. It is clear to see that the correlation among the data can be one of the key factors affecting the accuracy of the estimation, and we need some statistical structure to capture such correlation.

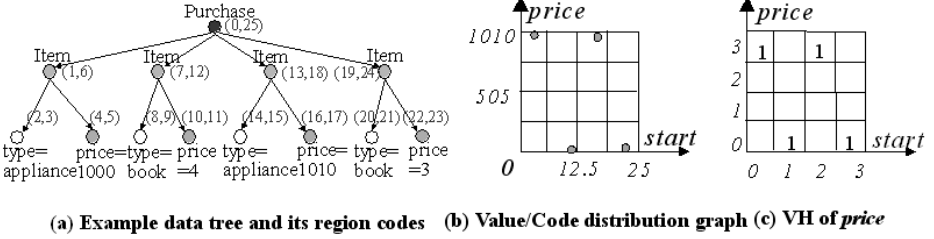


Fig. 1. An example data and the VH.

State of the art selectivity estimation methods for XML queries can largely be divided into two categories: the graph-based [3,9] and the histogram-based [6,12,13]. Graph based methods use a summary data structure, such as the DataGuide, to represent the occurrence frequency of the nodes and the paths in the tree structure. However, it cannot capture the correlation between different paths or nodes under the same root, while such information is essential in estimating the selectivity of branching queries with predicates. Histogram-based methods [12, 13] provide information about nodes' hierarchical distribution. Information about predicate selectivity is made available by storing the number of value nodes that have a given attribute value in a simple, one dimensional value histogram. But such selectivity is not associated with nodes' hierarchical distribution, in other words, we do not know how such selectivity will affect other nodes.

To obtain an accurate selectivity estimation using state-of-the-art histograms [12,13], we have to build a position histogram for each particular predicate, so that the position distribution reflects the existence of the predicate. However, it is impossible to build histograms for each possible predicate in path expressions.

In this paper, we propose to use a novel histogram, the *value histogram* (VH), to capture the correlation between the structure and the value. We define a set of operations on the *value histograms* as well as on the *code histograms* (CH) that keep track of the position distribution. We use these histograms to construct a cost tree. We can then compute the selectivity of any node in the path expression by evaluating the cost tree.

The rest of the paper is organized as follows. We introduce the *value histogram* (VH) in Section 2. In Section 3, we propose six basic operations that are used for selectivity estimation in Section 4. Section 5 shows the experiment result. We review related work in Section 6 and conclude in Section 7.

## 2 Value Histogram

In this section, we introduce the concept of *value histogram (VH)*. The 2-dimensional value histogram is devised to capture the distribution of predicate values across XML tree structures.

XML data are commonly modeled by tree structures. We encode each node  $n$  by a pair of integers ( $start$ ,  $end$ ) represents a region[14]. It is easy to tell how many descendants and ancestors  $n$  has by checking the regions containment. Fig. 1(a) shows an XML data tree whose nodes are labelled with region codes.

### 2.1 Value Histogram (VH)

The purpose of introducing *value histogram* is to associate the distribution of attribute values with the hierarchical positions of the nodes in the tree. We represent the relationship between value and position in a two-dimension space. Here, the x-axis corresponds to *start* (region code), and the y-axis corresponds to the *values* (for value nodes only). For each different node type, such as price (a continuous attribute), or type (a categorical attribute), we build a value/code distribution graph. Each price node in Fig. 1(a) maps to a unique point in the value/code distribution graph of Fig. 1(b). For example, point (10, 4) in Fig. 1(b) represents a node with  $start=10$ , and  $value=4$ , which is the second price node in Fig. 1(a).

To derive value histograms from the value/code distribution graphs, we discretize the region code *start* on the x-axis, and the continuous attribute values on the y-axis. More specifically, we discretize a two dimensional value/code distribution graph into  $g \times m$  grids. Fig. 1(b) is an example of  $g=4$ ,  $m=4$ . From a discretized value/code distribution graph, we derive the value histogram by counting the number of nodes inside each grid cell. Fig. 1 (c) is a  $4 \times 4$  value histogram for the *price* nodes. We use  $VH[i][j]$ , ( $0 \leq i < g$ ,  $0 \leq j < m$ ), to denote the number of *price* nodes whose value and *start* code fall in grid cell ( $i$ ,  $j$ ) in the value/code distribution graph.

The value histogram embodies the relationship between the value and the hierarchical structure, and reflects the value correlation of the data. We can build a value histogram for each type of the value nodes. Thus, a predicate in a path expression (e.g.,  $200 \leq price < 500$ ), together with a region constraint (e.g., descendants of a node whose region code is (7,28)), corresponds to a set of grid cells in the *value histogram*. We can sum up the counts  $VH[i][j]$  to get the distribution of the nodes that satisfy the predicate (see Section 3).

Note that in the value/code distribution graph and in the value histogram, we are only concerned with the *start* code but not the *end* code. This is because value nodes always appear as leaf nodes in the XML tree structure, and we only care about the distribution of its parent nodes and/or ancestor nodes. In order to count a node's ancestors and parents, we only need the *start* value of the region code.



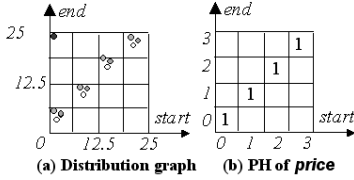


Fig. 2. Position Histogram.

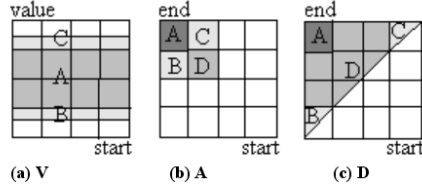


Fig. 3. The V, A, and D operations.

Beside *value histograms*, we need another type of histogram, the *code histogram*, to summarize the Ancestor-Descendent or Parent-Child relationships among the nodes.

## 2.2 Code Histogram

Several code histogram models [12,13] have been proposed in previous works to estimate the size of structure join. We modify some statistics and use them for selectivity estimation.

**Position Histogram (PH):** If we regard the region codes as coordinates in a 2 dimensional space, where values in the x-axis correspond to *start*, and values in the y-axis to *end*, the containment relationship between the nodes will be turned into the position relationship between the points. The points have the following property: (i) none of the points can fall into the area below the diagonal of the matrix, and (ii) given a point  $x$ , all the descendants of  $x$  appear to the right and lower part of point  $x$ , and all the ancestors of  $x$  to the left and upper part of point  $x$  [12,13]. Fig. 2(a) shows the distributions of nodes in Fig. 1(a). Fig. 2(b) shows a  $4 \times 4$  position histogram for the price nodes.

**PL Histogram (PLH):** Instead of using a 2-dimensional mapping, we can also map region codes into intervals in a 1-dimensional space [12]. Assume the root node is encoded (min, max). The PL histogram is constructed by dividing the range of [min, max] into  $g$  grids and counting the nodes in each grid.

The statistics we need is in addition to those used in [12]. For each grid, we need to know i) *the number of intervals that start in the grid (CD)*, ii) *the number of the intervals contained in the grid (CA)*, iii) *the average length of the intervals*, and iv) *the selectivity of the intervals*. We use these statistics in histogram operations in Section 3.

In general, we build code histograms for each type of node in the data tree to keep track of the structural distribution of the XML data. Next, we show how the histograms can be used for selectivity estimation.

## 3 Basic Histogram Operations

We use histograms for selectivity estimation of path expressions with predicates. In this section, we define six basic operations on histograms:  $V$ ,  $S$ ,  $D$ ,  $A$ ,  $PA$ , and  $PD$ . The first two,  $V$  and  $S$ , are used to study value correlation, and the rest are for hierarchical correlation.

**V: Value Selectivity Estimation.** The  $V$  operation counts the number of the nodes within a given *value range* using the value histogram. The result of the operation,  $SH[i]$  ( $0 \leq i < g$ ), is the number of nodes that satisfy the *value range* (in code grid  $i$ ).

Let  $P_{min} \leq attr \leq P_{max}$  be a predicate of interest. Assume the code/value space for attribute  $attr$  is divided into  $g \times m$  grids, with  $W_x$  and  $W_y$  being the x-axis and y-axis grid width respectively. Then the number of nodes in each code grid that satisfy the predicates is estimated to be:

$$\begin{aligned} SH[i] = & \sum_{j=\lfloor P_{min}/W_y \rfloor + 1}^{\lfloor P_{max}/W_y \rfloor - 1} VH[i][j] \\ & + (\lfloor P_{min}/W_y \rfloor + 1 - P_{min}/W_y) VH[i](\lfloor P_{min}/W_y \rfloor) \\ & + (P_{max}/W_y - \lfloor P_{max}/W_y \rfloor) VH[i](\lfloor P_{max}/W_y \rfloor) \end{aligned} \quad (1)$$

There are three additive terms in the formula. The first adds up the grids fully included in the given value range of the predicate, and the other two adds up the grids partially included (assuming uniform distribution). They are shown by the shaded area in Fig. 3(a).

**S: Value/Code Selectivity Estimation.** The  $S$  operation (Eq 2) estimates the number of nodes in each position grid satisfying a predicate. It associates the value distribution (Eq 1) with the hierarchical distribution (position histogram). The result of the  $S$  operation is a two dimensional *code histogram* ( $CH$ ). The  $S$  operation assumes that the selectivity of a value on a node in a certain grid has uniform impact over the nodes in the same position grid. Our experimental results in Section 5 show that the assumption works well.

$$CH[i][j] = PH[i][j] \frac{SH[i]}{\sum_{j=0}^{g-1} PH[i][j]} \quad (2)$$

**D: Descendant Selection** and **A: Ancestor Selection.** Given a node, the  $D$  operation ( $A$  operation) returns the distribution of its descendants (ancestors) in a *code histogram* ( $CH$ ). The meaning of the two operations is illustrated by the position histograms in Fig. 3(b) and (c). For instance, to find the descendants for a node in region “A”, we include the region of “D”, and partially include the region of “B” and “C”. Similar reasoning applies to the  $A$  operation. Due to a lack of space, we refer readers to [15] for a detailed description of the computation of  $D$  and  $A$ .

**PD: Descendant Pruning** and **PA: Ancestor Pruning.** Given a node, we might be only interested to know the number of its direct children instead of all of the descendants. However, such statistics is not immediately available from the position histogram. The  $PD$  and  $PA$  operations are used to exclude the descendants by estimation. Due to a lack of space, we refer readers to [15] for a detailed description of the computation of  $PD$  and  $PA$ .

## 4 Selectivity Estimating of Path

In this section, we demonstrate how to construct a cost tree using the operations discussed in Section 3 for selectivity estimation. The original value histograms

(*VHs*) and code histograms (*CHs*) are used as input, and the output is a code histogram that maintains the distribution information of the node satisfying the path.

#### 4.1 Simple Predicate and Simple Path

To estimate the selectivity of a simple predicate, for example, [*type*=*'appliance'*], we compute *SH* by applying the *V* operation (Eq 1) on the value histogram built for the value nodes. Then we perform the *S* operation on the *SH* and the code histogram for the *type* nodes to get a new code histogram *CH*, which includes in each grid only the nodes satisfying the predicate.

A simple path can have two basic types of components: *a/b* and *a//b*. For *a//b*, it is easy to estimate the distribution of *a* or *b* satisfying path *a//b*. We simply use the *A* or *D* operation.

It is not as straightforward to estimate the distribution of *b* satisfying the path *a/b*, because *b* nodes might be self-nested and we must exclude those *b* nodes that are descendants of *a* but not children of *a*. To achieve this goal, we first perform the descendent pruning operations (*PD*). Then, we run *D* operation to get the estimation. Similarly, to estimate the distribution of *a* nodes satisfying path *a/b*, we perform the ancestor pruning operation (*PA*).

#### 4.2 Complex Path with Predicates

There are two factors affecting the selectivity of a node in a path: one coming from the ancestor path, the other from the descendant paths. We discuss the two factors respectively.

To estimate the selectivity of the node satisfying the ancestor path, we perform a series of *D* operations on the code histograms of the nodes in the ancestor path. The result of one operation is used as input to the next operation until we derive the code histogram of the target node in the path. If the relationship between two nodes is *'/'*, we need additional *PD* operations as well. For example, assume we want to compute the selectivity of node *n3* in path expression *n1/n2//n3//n4[a>v]*. Because the relationship between *n1* and *n2* is *'/'*, we perform a *PD* operation on the code histogram of node *n2* to prune those nodes that are descendants of *n1* but not children of *n1*. Then, we run a *D* operation on the result of the *PD* operation and the code histogram of node *n1*. Another *D* operation is needed to acquire the selectivity of node *n3*. Because the relationship between *n2* and *n3* is *'//'*, the *PD* operation is skipped. To estimate the selectivity of the nodes satisfying the descendant path, we perform a series of *A* operations along the path in the reverse direction. If the relationship between the nodes is *'/'*, we perform the *PA* operation on the parent nodes and the *PD* operation on the children nodes. In our example, the descendant path of *n3* is *//n4[a>v]*. We deal with the predicate first by performing a *V* operation and an *S* operation. Because the relationship between *n4* and *a* is *'/'*, before the *A* operation, we perform a *PD* operation on the resulting code histogram of the *V* operation and a *PA* operation on the code histogram of *n4*. The cost tree is shown in Fig. 4(a).

Any complex path can be decomposed into a set of simple predicates and simple paths. If the selectivity of a node is affected by many paths, we compute the selectivity of each path, and combine them together. For example, in path  $n1[n2 > v \text{ and } n3 > w]//n4$ ,  $n1$  is a switch node, which has three descendant paths: predicate  $n2 > v$ , predicate  $n3 > w$ , and a descendant path  $//n4$ . We deal with them one by one. First, we get the selectivity of  $n1$  from the predicates, and then use the intermediary results to compute the selectivity from the descendant path. The cost tree is shown in Fig. 4(b).

Sometimes the computation can be complicated. In our previous example, the path expression has 5 nodes, but the cost tree in Fig. 4(a) for selectivity estimation has 9 operations. However, if we know some nodes are not self-nested, we can omit some. For example, we can omit the *PD* and *PA* operations in Fig. 4(a), if we know (from the schema) that all  $n3$  nodes are descendants of  $n2$ , all  $n2$  nodes are children of  $n1$ , and all  $a$  nodes are children of  $n4$ . The simplified cost tree is shown in Fig. 4(c).

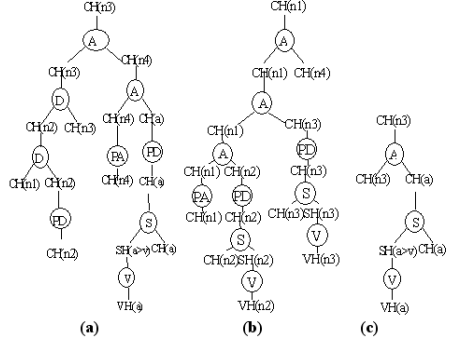


Fig. 4. Complex Cost Trees.

## 5 Experiments

**Data Sets:** We use two XML data sets in our experiments. One is the Xmark data set, whose sizes can range from 100KB to 100MB. The data schema contains 550 different types of nodes and 10 of them are value nodes. The other dataset, Bib, which is generated by XML SPY, has a simple structure, but the structure and the values in the data are highly correlated.

**Storage Cost:** The size of the statistical information we need is proportional to the number of nodes in the data schema as well as to the number of grids in the histogram. Since the schema is much smaller than the data, major savings can be realized in our approach. The storage cost of the *value histogram* (*VH*) is  $g \times m \times n \times \text{gridsize}$ , where  $g$  is the number of the code grids,  $m$  is the number of the value grids, and  $n$  is the number of value nodes. The storage cost of the *position histogram* (*PH*) is  $g^2 \times n \times \text{gridsize}$ . For example, an Xmark document with 550 nodes in schema, the total histograms size is 675KB. The *VH* and *PH* are sparse matrix that can be compressed. In Fig. 5, *CVH* represents the size of the compressed *VH*.

**Estimation Cost:** We use six typical queries with different selectivity. The query time is sensitive to the size of data sets, but the estimation time is not, instead, it is sensitive to the size of the schema, the size of the histograms, and the number of operations in the cost tree. We use a 2M data set to evaluate the queries. Fig. 6 shows the average estimation time and query time. The estimation

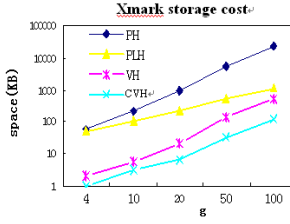
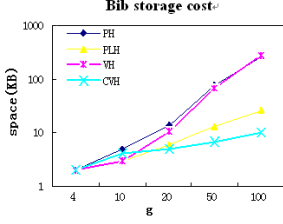


Fig. 5. Storage cost of statistical information.

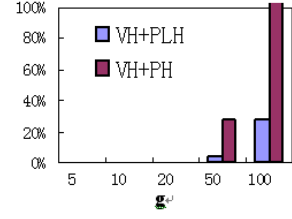


Fig. 6. Estimation Cost.

time for histograms with fewer than 50 grids is very short (less than 1 millisecond). With the increase of the grids, the estimation time of *PH* method increases faster than *PLH* method. When the grid is as fine as  $100 \times 100$ , estimation based on *PH* takes more time than query.

**Estimation Accuracy:** Fig. 7 shows the estimated accuracy for the above queries using *PLH+VH* and *PH+VH* methods. Note that the first method has very reasonable error margins ( $<10\%$ ) when the number of grids are greater than 20. For some queries, for example, queries Q1, Q2 and Q3, the *PLH+VH* method has virtually no error.

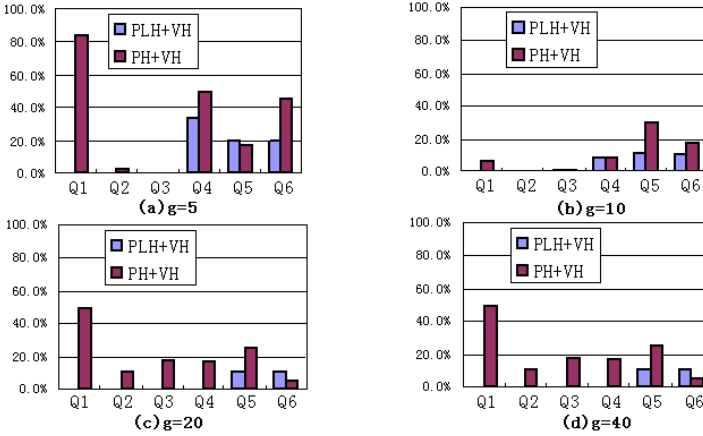


Fig. 7. Estimation Accuracy.

We also note in Table 1 that the uniform and independent distribution assumption results in large errors. An important factor that contributes to these errors is the correlation between the structure and the value in the data set. For example, query Q5 finds `close_auction` with `quantity=1` and `price >100`. There are 80% of the `quantity` nodes whose value is 1, and 80% of the `price` nodes whose value is greater than 100. However, few `close_auction` nodes have both attributes. But with the uniform and independent distribution assumption, the total selectivity will be 60%. The errors will be magnified with longer paths and more predicates. The errors of the *PH+VH* method are caused by the grid. In the sample data sets, the code region is wider than  $[0, 10000]$ . The width of each

**Table 1.** Example queries list.

	<b>Acutnal</b>	<b>Uniform</b>	<b>Queries</b>
Q1	60%	50%	//bib[No.>5]
Q2	40%	5%	//Book[price>100]
Q3	10%	1%	//Book[author='Mike']
Q4	80%	40%	//Paper[type=1][page>4]
Q5	5%	60%	//close_auction[quantity=1][price>100]
Q6	30%	50%	//Open_auction[increase>20]

grid is as large as 100 even for 100\*100 grids. The difference between the code of a parent code and its children may be very small (sometimes only 1). Thus, they very likely appear in the same grid. Estimation in the same grid is based on the uniform assumption, which leads to the error. This type of error is not evident in the *PLH+VH* method because *PLH* counts the ancestors and the descendants independently.

## 6 Related Works

McHugh et al [9] used DataGuide to maintain statistics information. The system stores selectivity information for all paths in the database of length up to  $k$ , and uses it to infer selectivity estimates of longer path queries. Markov table is used to maintain statistics of all paths up to a certain length [1], and such information can be aggressively summarized. These techniques do not maintain correlations among paths, and consequently, they do not support accurate estimation of path expressions with embedded predicates.

Chen et al [3] used pruned suffix trees to summarize tree structures, and used a set hashing technology to capture the correlations among the various sub paths in the data. However, the technique only applies to specific twig patterns which contain only parent-child links and the predicate can only have the equality test. Another problem of the method is that their correlation computation is based on the same data type.

StatiX [6] combines schema information with histograms. The system gives each node an id, and ranges of ids of different types do not overlap. For each node type, histogram is used to store the count of children nodes. This method can estimate the selectivity of children nodes, but not the selectivity of the ancestor nodes satisfying some descendant paths.

Timber [13] combined the XML data encoding with a 2 dimension histogram method, using a position histogram to collect and estimate the sizes of containment join. The computation needs the position histogram of both the ancestor set and the descendant set. In a path expression, the predicate may be very complex. It is impossible to build position histograms for each set of predication. Furthermore, the algorithm can only estimate the selectivity of ancestor-descendant join. However, there can be many parent-child relations in path expression queries. Wang et al [12] provided a one-dimension histogram method for containment join size estimation, which improves accuracy in some cases, but the computation limitation also exists.

## 7 Conclusion

In this paper, we propose to use a novel type of histogram to capture such correlations. Through a set of operations on the histograms, we are able to estimate the selectivity of any node in a complex path expression. Experimental results indicate that our method provides improved accuracy especially in cases where the distribution of the value or structure of the data exhibit certain a correlation. Our future work will extend the method to support XQuery.

## References

1. Aboulmaga, A. R. Alameldeen, J. Naughton. Estimating the Selectivity of XML Path Expressions for Internet Scale Applications. In: Proc. of the 27th VLDB Conf., Italy, 2001. 591-600.
2. Chan, P. Felber, M. Garofalakis, R. Rastogi. Efficient Filtering of XML Documents with Xpath Expressions. In Proc. of ICDE Conf., 2002. USA. 235-244.
3. Z. Chen, H. V. Jagadish, F. Korn, N. Koudas, S. Muthukrishnan, R. Ng, D. Srivastava. Counting Twig Matches in a Tree. In: Proc. of ICDE Conf, 2001. 595-604.
4. Choi, M. Fernandez, J. Simen . The XQuery Formal Semantics: A Foundation for Implementation and Optimization. Technical Report, <http://www.cis.upenn.edu/kkchoi/galas.pdf>, May 31, 2002.
5. B. Cooper, N. Sample, M. J. Franklin, G. R. Hjaltason, M. Shadmon. A Fast Index for Semistructured Data. In Proc. of 27th VLDB Conference, Roma, 2001, 341-350.
6. J. Freire, R. Jayant, M. Ramanath, P. Roy, J. Simeon. StatiX: Making XML Count. In: Proc. of ACM SIGMOD 2002, USA. 181-191.
7. V.H. Jagadish, S. Al-Khalifa, L. Lakshmanan, A. Nierman, S. Paparizos, J. Patel, D. Srivastava, Y. Wu.. Timber: A native XML Database. Technical report, University of Michigan, 2002.
8. Q. Li, B. Moon. Indexing and Querying XML Data for Regular Path Expressions. In: Proc. of 27th VLDB, Roma, Italy, 2001. 341-350.
9. McHugh, J. Widom. Query Optimization for XML. In: Proc. of 25th VLDB Conference, UK, 1999. 315-326.
10. T. Milo and D. Suciu. Index structures for path expression. In Proc. of 7th International Conference on Database Theory, 1999. 277-295.
11. N. Polyzotis, M. Garofalakis. Statistical Synopses for Graph-Structured XML Databases, In: Proc. of ACM SIGMOD Conf., 2002. 358-369.
12. W. Wang, H. Jiang, H. Lu, J.F. Xu. Containment Join Size Estimation: Models and Methods. In: Proc. of ACM SIGMOD 2003, USA. 145-156.
13. Y. Wu, J. Patel, H. Jagadish. Estimating Answer Sizes for XML Queries. In: Proc. of EDBT 2002. 590-608.
14. Zhang, J.F. Naughton, D.J. DeWitt, Q. Luo and G.M. Lohman. On Supporting Containment Queries in Relational Database Management Systems. In: Proc. of ACM SIGMOD Conf., 2001. 425-436.
15. Yu Wang, Xiaofeng Meng, Shan Wang. Estimating the selectivity of PWP by histograms. Technical Report. <ftp://202.112.116.99/pub/paper>.

# Wrapping Web Pages into XML Documents

Tao Fu

School of Computer Science, Carleton University  
1125 Colonel by Drive  
Ottawa, Ontario, Canada K1S 5B6  
tfu2@scs.carleton.ca

**Abstract.** XML is gaining popularity as an industrial standard for presenting and exchanging structured information on the web. Meanwhile, the majority of documents on-line are still marked up with HTML, which are designed specifically for display purposes rather than for applications to automatically access. In order to make web information accessible to applications so as to afford automation, inter-operation and intelligent services, some Information Extraction programs, called “wrappers”, have been developed to extract the structured data from HTML pages. In this paper, we present a layout-based approach to separate the data layer from its aspect of presentation in HTML and extract the pure data as well as its hierarchical structure into XML. This approach aims to offer a general purpose methodology that can automatically convert HTML to XML without any tuning for a particular domain.

## 1 Introduction

In order to make web information accessible to applications, some Information Extraction (IE) programs, called “wrappers”, have been developed to automatically extract information from a (usually not well-structured) format and maps it to a structured format. In context of web extraction, the mapping is usually carried out from HTML to XML.

Previous work can be classified into two categories, depending on whether the HTML input is regarded as a sequential character string, such as TSIMMIS [9], Editor [3], FLORID [4] and DEByE [6]; or a pre-parsed document tree, such as W4F [2], XWrap [7] and Lixto [5]. Basically, those approaches requires either heavy human-annotated web pages as training examples or heavy hand-coding to generate a specialized extractor for each web data source.

In this paper, we propose novel approach to automatically extract structured data from HTML pages without any tuning for a particular domain. This method doesn't require any prior human intervention. It uses features derived directly from the layout of HTML pages. The motivation is lying on the nature of HTML: a semi-structured document with data and its presentation tags.



## 2 Models and System Overview

### 2.1 Transitional Document Object Model

According to the DOM specifications, the HTML/XML document is defined as an ordered tree of element nodes, to which attribute and text nodes are attached. For example, the Yahoo page displayed as in Figure 1 can be represented in HTML DOM tree like Figure 2 (a). If we convert it to XML document, we have the XML tree as shown in Figure 2 (b).



Fig. 1. A partial view of Yahoo's homepage.

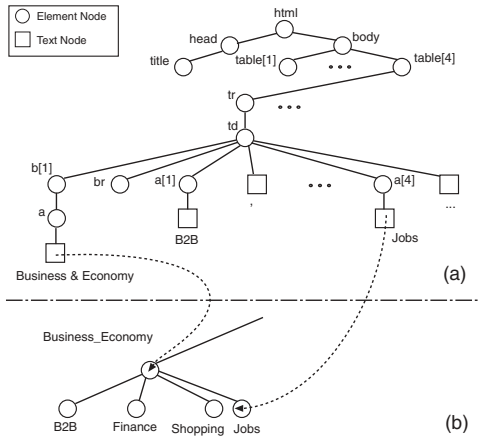


Fig. 2. Node-2-node transforming between the HTML DOM and XML DOM.

Comparing with these two DOM trees, we can see that the XML DOM describes the structured data itself while the HTML DOM focuses on the representational logic and leaves all the actual data lying on the bottom of the tree. Despite the fact that one is focused on data structure while the other is focused on data representation, XML and HTML have one common DOM core. From this point of view, the conversion

from HTML to XML is intuitive and as simple as node-to-node matching. However, the fact is that, due to the diversity of HTML representation and constantly changing web sources, mapping the HTML nodes directly to XML nodes is difficult and inefficient. To ease this complicated conversion process, we build a middle-tier model, called Transitional DOM (T-DOM). It works like a well-organized bookshelf, where the HTML nodes are re-arranged so that the wrapper program can select and map them to XML nodes more easily.

## 2.2 Atomic Units of Web Information

The task of information extraction is the focused searching for *meaning* in the target documents, where meaning is understood in terms of *facts*, formally described as a fixed set of lexical objects [1]. In the context of HTML documents, these lexical objects are atomic units of web information appearing in a specific region of the browser's display window. In order to detect and access these information units in our Transitional DOM, we introduce the definition of *record* and *delimiter* which are both extended from the HTML nodes.

**Definition 21** A record is an HTML node which has a meaningful text value. All the records in an HTML DOM are denoted as  $\mathbb{R}$ . A delimiter is an HTML node which does not have a meaningful text value. All the delimiters in an HTML DOM are denoted as  $\mathbb{D}$ .

**Remark 21** We give the following common notes for the Definition 21:

1. For languages that are based on ASCII, ISO 8859 or EUC standards, like English, the meaningful text value is a string containing any characters from  $[0\dots9][a\dots z][A\dots Z]$ . For other human languages, the definition of a meaningful text value may vary, but the basic criteria is that every record must carry a fact of web information.
2. The definition of record is upward transitive in a dominance relation, such that, if a node is a record, then its parent node is also a record. In contrast, the definition of delimiter is downward transitive in a dominance relation, such that if a node is a delimiter, all its child nodes are delimiters. In other words, if all the child nodes are delimiters, then this node is also a delimiter.
3. Most attributes in HTML, such as *border*, *width*, *onClick* and *onLoad*, are layout or event properties that contain no facts of web information to extract. So, we assert that attribute nodes, except *href* and *alt*, are not considered as records even if they have a meaningful text value.
4. Both records and delimiters are extended from the interface of the HTML DOM and inherit all the properties of DOM nodes, such as *label*, *value* and *level*.

As shown in Figure 2, element nodes *b1*, *a1* and *a4* are records, since they have text values "Business & Economy", "B2B" and "Jobs" respectively. Aside from text nodes, a text value could also be obtained from an attribute node *alt* or *href*. An *img* node, with HTML source ``, is a record because *alt* has a meaningful text value "Yahoo! Travel". The element node *br* and the text nodes *“,”* and *“...”* are delimiters since they have no meaningful text value.

### 2.3 Hierarchical Relations of Web Data

A single record in the HTML DOM, just like a single word in an article, still lacks meaning without context. To explore the hierarchical relations between records, we need to group the consistent records into content blocks first, and then find the topics in each consistent record.

**Definition 22** *The notions of content blocks and topics are defined as follows:*

1. A content block, denoted as  $\psi$ , is a set of adjacent records on the same level in the DOM tree.
2. A topic node (or topic), denoted as  $\tau$ , is a record which has the property of being the subject of other records in the same content block.

**Remark 22** *We give the following common notes for the Definition 22:*

1. The topic is extended from the interface of record and inherit all the properties of DOM nodes, such as label, value and level.
2. A content block should not be confused with the basic component NodeSet in a DOM tree. There are three essential differences. First, the NodeSet is a set of nodes of all types, including records and delimiters, while in a content block, all nodes are records. Second, nodes in a NodeSet can be on different levels in the tree, and they may not be adjacent, while in a content block, all nodes must be adjacent and on the same level. Last, nodes in a NodeSet can have different parent nodes in the HTML DOM tree, while all of records in a content block must have the same parent. For this property, we can denote it by a sub-tree in Transitional DOM tree, such as  $T(x)$ , where  $x$  is the common parent of all records in the content block.
3. If  $x_1, \dots, x_n$  are adjacent records on the same level in the DOM tree, a set  $\{x_1, \dots, x_n\}$  is a content block constructed from records  $x_1, \dots, x_n$  using a set constructor of order  $n$ .
4. If a content block  $\psi$  is  $\{x_1, \dots, x_n\}$ , a tuple  $\langle \tau_1, \dots, \tau_k \rangle$ , where  $(\tau_1, \dots, \tau_k) \in \psi$ , is a list of topics constructed from records  $x_1, \dots, x_n$  using a tuple constructor of order  $k$ .
5. All topics come from the records of content blocks. We give a heuristic that every content block in the HTML DOM contains a default topic. In case no explicit topics can be found, we can add a certain record into the content block to be its default topic.

Topics are obvious on a browser's window as shown in Figure 1. This is because the author of the web document uses many specific HTML tags and structures to *guide* a browser in emphasizing these topics so that they look different from other representations. A wrapper program can use a similar method to abstract the subject of other records in a content block.

## 3 Content Partition

Whether it is manually composed or generated automatically by an application, a web page commonly keeps consistent information as discerned in the same content block,

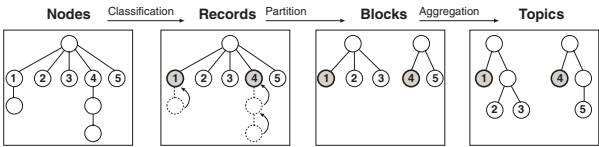


Fig. 3. Transforming functions for building the Transitional DOM.

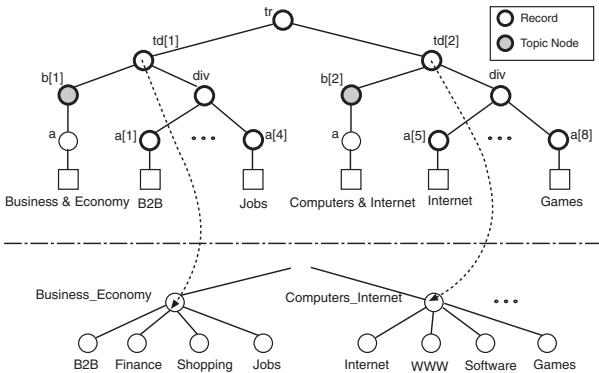


Fig. 4. Mapping the Transitional DOM to the XML DOM.

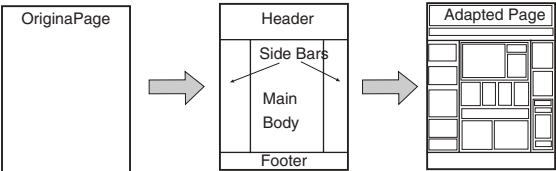


Fig. 5. Content Blocks Identification.

and different topics are separated by apparent visual boundaries. That is what the *Layout* of a web page means.

The purpose of content partition is to recover these information blocks based on the clues that authors embed in the layout information. Figure 5 shows how the process of identifying these visual cues is performed in an iterative way. It starts with the whole web page as a single content block (made up of smaller content blocks), and then iteratively partitions each content block into smaller ones until the process reaches the final solution.

4 Topic Aggregation

Normally, a topic node conforms to the following two characteristics that allow it to compare its peer records in the same content block. First, a topic node is a record with

emphasis at a preceding position compared with its sibling records in the same block. The main strategies for emphatic purpose are 1) Using a distinguishing style, such as bold fonts. 2) Placing an indication symbol, like a colon “:”, to infer dominance. 3) Creating an indentation between primary and secondary. Second, topic nodes have the most similar properties and a maximal frequency of occurrence in the same content block.

**Example 41** An HTML block displayed as “**Tel:**, 2280256, **Fax:**, 2280255, ...” has the following HTML codes:

```
<p>
  <b>Tel: </b> 2280256 <b>Fax: </b> 2280255 ...
</p>
```

The bold style nodes (**Tel:** and **Fax:**) are topic nodes of the block  $\{b, 2280256\}$  and block  $\{b, 2280255\}$  respectively. Compared with its peer records in the same block, a topic node is a leading record with significant text value, and for emphatic purposes, they are commonly in bold style. Moreover, the repetition of a similar structure  $\{b, s\}$  has a maximal frequency of occurrence in the content block, where  $s$  is a text node.

In the HTML specification, the description of one topic may intertwine with the descriptions of some other topics. For example, the descriptions of two topics in the HTML source may follow the sequence part1 of topic1, part1 of topic2, part2 of topic1, part2 of topic2. Thus, the descriptions of both topic1 and topic2 are not contiguous. However, when they are displayed on a Web browser, they appear contiguous to human viewers. This leads to a more complicated hierarchical relation hidden inside HTML DOM: two-dimensional space. The only case of two-dimensional space appearing in HTML documents is  $m \times n$  tables.

Table representation could be very complicated as shown in the Figure 6 (a). This was due to the two specific attributes of `<td>` nodes: 1) *rowspan* =  $i$  can span the `<td>` cell  $i$  rows; 2) *colspan* =  $j$  can span the `<td>` cell  $j$  columns. Since these variations break the convention of hierarchical representation when compared with a normal HTML table, we need a procedure to recover the normal hierarchies implied by these span attributes.

**Example 42** Figure 6 (a) is a two-dimension space with *colspan* and *rowspan* at cells. To normalize this table, we expend the cells with *colspan* and *rowspan* in columns and rows firstly, as shown in Figure 6 (b), and then merge the first two rows since they contain the topic nodes, as shown in Figure 6 (c).

After these complicated tables are normalized, we can reduce the two-dimensional table to a set of one-dimensional spaces, as shown in Figure 7, and then use the same approaches developed for one-dimensional space to construct the hierarchies.

## 5 Semantic Annotation

In this section, we provide an automatic annotation algorithm which is motivated by the nature of our Transitional DOM. After the content partition and topic aggregation processes, we have transformed the original HTML DOM into the T-DOM tree where the document structure is enhanced using topics and blocks. The topic nodes in content



**Fig. 6.** The process of table normalization.

blocks intuitively act as good annotators that can build the hierarchical relations. Given highly hierarchical input, most semantic markup can be done quickly using the topic nodes as the XML elements. For less-hierarchical pages, we can add some text mining approaches, such as regular expression, to improve annotation accuracy.

## 5.1 Annotation by Topic Nodes

The semantic annotation is a process of building the XML DOM tree recursively. First, we give two trees represented by two roots, one for XML DOM the other the HTML source. In our system, this HTML source is actually the T-DOM we have obtained from the original HTML source. Then, we decide the types of the HTML root: if it is a leaf record without child records, we just append it to the XML root and end the process; if it is a T-DOM block that contains child records but without a specific subject, we keep the original XML root and send each of the child records iteratively to the beginning of the algorithm and run the process recursively; if it is a T-DOM topic that contains a topic node as the first child node, we construct a new XML node according to this topic node and append it as the child node of the XML root, passing this new XML node as the new root of XML DOM, as well as each of the other child records of the T-DOM node to run the algorithm recursively.

An example of the semantic annotation is given in the section of system overview. Figure 4 shows the mapping from T-DOM to XML DOM, and Figure 6 shows the result of using topic nodes (like *Business & Economy*) to conduct the hierarchical structure and gain the XML output.

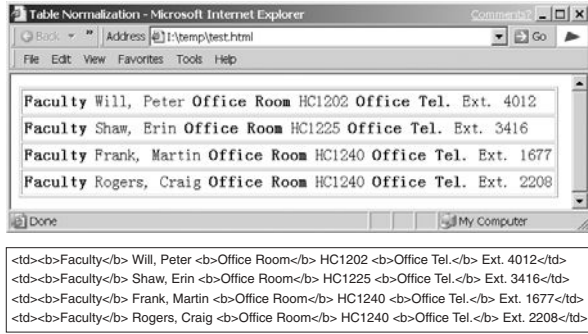


Fig. 7. Reducing the two-dimensional table to a set of one-dimensional spaces.

## 5.2 Annotation by Regular Expressions

Given highly hierarchical input, most semantic markup can be done quickly by mapping the topic nodes to the XML elements. For less-hierarchical pages, we can add some text mining approaches, such as regular expression, to improve the annotation process.

Regular Expression (RE) is the simplest NLP (Natural Language Processing) technique that can be used to extract structured data from snippets of unstructured text, such as time and phone number.

**Example 51** A set of street addresses: 1020 Bank St., 512 Oak Blvd., 416 Main St. and 97 Adams Blvd, all start with a pattern (Number + Capitalized Name) and end with (Blvd.) or (St.).

We refer to the starting and ending patterns together as the regular expression:

```
<address match = "\d? \s[A-Z].*\s[(Blvd)|(St)].*" type = "re" />
```

## 6 Experiments

Based on this system structure, we have implement a web application called “XML Gateway”<sup>1</sup> which can convert any HTML documents into XML files on-the-fly. We use this prototype system to test our approaches and provide the experimental results.

### 6.1 Design and Implementation of XML Gateway

XML Gateway has been written in J2EE and implemented on Apache Web Server. In order to make our extractor easy to use, we implemented it as a web proxy. This allows an administrator to set up the extractor and provide content extraction services for a group. The proxy is coupled with a graphical user interface (GUI) to customize its behavior. The current implementation of the proxy is in Java for cross-platform support.

<sup>1</sup> The XML Gateway web application can be reached at <http://tao.my-net-space.net/h2x>

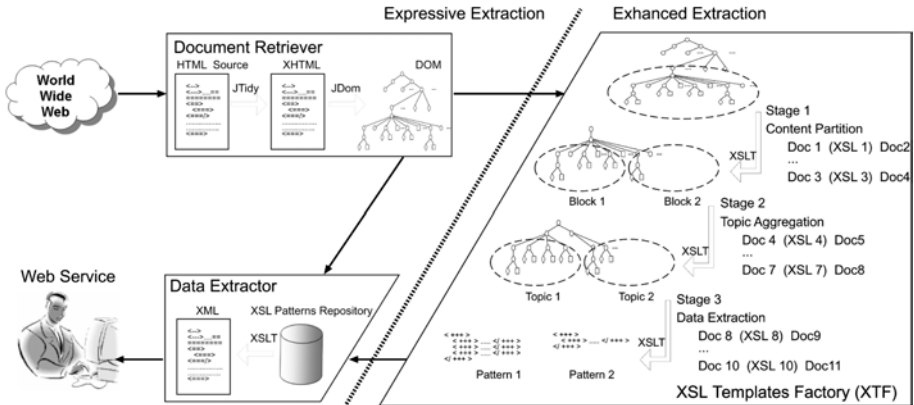


Fig. 8. System architecture of the XML Gateway wrapper system.

Figure 8 is the system architecture of “XML Gateway”. There are three sub-systems under the main-frame of the whole system: *Document Retrieving Module*, *Data Extraction Module* and *XSL Templates Factory Module*. These sub-systems work together in two extraction modes ( *Expressive* and *Enhanced* extraction). In the expressive mode, the extraction patterns are retrieved directly from pattern repositories and used for mapping XML with the HTML DOM trees. If the extraction patterns are not available or the old patterns can not extract the right data, the HTML DOM is passed into an enhanced mode, where a set of structure and semantic syntheses are processed to find the extraction patterns.

## 6.2 Evaluation

Information extraction standards of measurement grew out of Information Retrieval (IR) metrics of *Recall* and *Precision*, but the definitions of these measurements were altered. In the IE task, recall can be interpreted as a measure of the records that has been correctly extracted, and precision as a measure of the extracted records that is correct. Note that the correct record does not mean we have *incorrect* record on web pages. A correct record means a meaningful information fraction that can be part of a specific topic. Both recall and precision are always on the interval  $[0, 1]$ , their optimum being at 1.0. They are, however, inversely related to each other, meaning that by allowing for a lower recall you can achieve a higher precision and vice versa.

We selected 20 different style and change period web pages to test our solution and give the results of average recall is 0.9 and the average precision is 0.92. Throughout the experimental evaluations, we show that our layout-oriented method leads to a highly automatic and accurate transformation for the extraction of semi-structured documents on the web.



## 7 Conclusions

Traditional information extraction techniques are based on *full-text* and treat the web document as *a bag of strings*. They either ignore the markup tags or keep them as delimiters of strings. The limitation of these approaches is that a large part of structural information, that the authors are trying to indicate by the layout of information, has been discarded or ignored. For example, `<b><a> ... </a></b>` and `<a><b> ... </b></a>` look the same in a browser but have different DOM structure. Traditional wrappers construct two different rules for the task which is laborious and inefficient.

Our approach is also less labor-intensive than other approaches that manually or semi-automatically generate wrappers, and it is generally insensitive to changes in web pages format. Instead of generating wrappers with respect to specific HTML page contents, we generate wrappers based on recognizable data layouts in HTML semantic logics. This significant departure from past approaches is a key difference in what we present here.

## References

1. E. Agichtein and L. Gravano, "Snowball: Extracting relations from large plain-text collections," in *Proceedings of the Fifth ACM International Conference on Digital Libraries*, 2000.
2. F. A. Arnaud Sahuguet, "Web ecology: Recycling html pages as xml documents using w4f," pp. 31–36, WebDB, 1999.
3. P. Atzeni and G. Mecca, "Cut and paste," ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, 1997.
4. G. L. W. M. B. Ludascher, R. Himmeroder and C. Schlepphorst, "Managing semi-structured data with floris: A deductive object-oriented perspective," pp. 23(8):1–25, Information Systems, 1998.
5. R. Baumgartner, S. Flesca, and G. Gottlob, "Visual web information extraction with lixto," in *The VLDB Journal*, pp. 119–128, 2001.
6. F. Laender, B. Ribeiro-Neto, and A. da Silva, "Debye - data extraction by example," pp. 40(2):121–154, Data and Knowledge Engineering, 2002.
7. L. Liu, C. Pu, and W. Han, "XWRAP: An XML-enabled wrapper construction system for web information sources," in *ICDE*, pp. 611–621, 2000.
8. W3C. Extensible Stylesheet Language (xsl), version 1.0, Recommendation 15 October 2001.
9. H. G. M. Y. Papakonstantinou, A. Gupta and J. Ullman, "A query translation scheme for rapid implementation of wrappers," pp. 97–107, International Conference on Deductive and Object-oriented Databases (DOOD'95), 1995.

# Rapid Prototyping for Web Applications

Chee Chern Lim, Man Hing Yu, and Jesse J. Jin

School of Information Technologies, F09  
University of Sydney, NSW 2006, Australia  
{chern, zenoy, jesse}@it.usyd.edu.au

**Abstract.** This paper proposes a web-based toolkit for web page developers to produce functional version of database-driven web systems. Many researchers realize the significant difference between the development process of web systems and conventional software systems. A key difference in web development is the clients understood their requirements poorly at the beginning of the project. The objective of this toolkit is to achieve rapid software prototyping in building web application. This result will shorten the timeframes of the actual development process, and developers can produce a functional working system with a minimum amount of effort. The functional system not only will help the clients to have better understanding of the project requirements, but also articulating other possible solutions to their problems. This paper outlines the development of today web applications, and how the toolkit is designed to improve the development of web-based information systems.

## 1 Introduction

Since the World Wide Web has increasingly gained acceptance as a medium for information exchange, the development of WWW applications has become more and more important in recent years. With the Internet dominating the development of information communication technology, web-based systems solutions are becoming increasingly attractive to business community. The major advantages of the Internet-enabled applications include platform independence and interoperability. As the web emerged becoming a global information network, web applications development method has transformed from simple static information displaying to dynamic web page generating, and now to an in-depth complex functionality of database-driven applications. These applications are the solutions to cover the variety of web system from the electronic commerce to the information provision as well as the management of business-to-business supporting systems.

In today web development, portal technology has played a significant role in most Internet applications systems. A portal is defined as a single, integrated point of access to information, applications and people. Portals are largely based on existing web application technology and can be categorized into public portals, enterprise portals, marketplace portals or specialized portals by according to the users they serve and the services they offer. Most current portal server products are Java/J2EE-based. The main competitive portals include the Apache Jetspeed Portal Framework, IBM WebSphere Portal Server and BEA WebLogic Portal [1].

In recent years, many researchers have realized the significant difference between the web system development and conventional software development [2], [3], [4]. Researchers recognize that web systems have various unique characteristics that are poorly addressed by conventional development practices. These characteristics include the technical differences and organizational differences. Both differences are significant crucial, however a number of organizational characteristics has shown its uniqueness in web systems and proven to be possibly more important than characteristics in the technical differences. Among these factors, the most significant of these characteristics is the client uncertainty. They found that in the web system development the client having difficulty in articulating the system requirements at the commencement of the project. This is due to the clients' poor understanding of their own needs with respect to the web technology [5].

In order to further explore the issues around web development processes and verify the client uncertainty characteristic, same group of researchers undertook an extensive set of industry interviews and surveys. The surveys conclude that the clients have a strong feeling that they did not understand their own needs. They have an initially very poor understanding of their requirements and this evolves during the product development. Consequently, this uncertainty characteristic causes the frequency of content or requirement changes as the project being developed; this effect dramatically increases the actual costs of the project. In addition, many web projects are vision-driven rather than requirements-driven. This again results to an initial lack of clarity of the actual system functional requirement. To resolve the client uncertainty in web system, the researchers have propose a generic web development process that is both iterative and utilizes design prototypes to assist the client in articulating possible solutions and thus formulating requirements. Throughout the process cycle, developers will repeatedly build prototypes functional system and explore them with the client to obtain feedback and distill from this information on the client needs [6]. Rapid prototyping in web system can reduce the client uncertainty. Consequently, this leads to an increasing importance of the incremental and prototyping approaches.

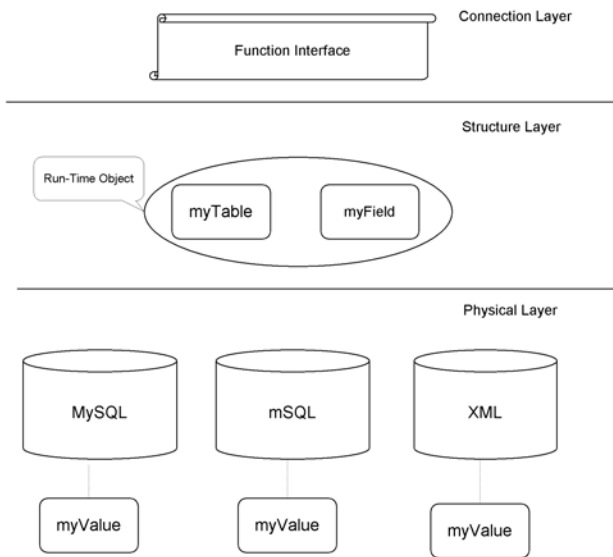
A software rapid prototype is a dynamic visual model providing a communication tool for customer and developer that are far more effective than either narrative prose or static visual models for describing functionality. Prototyping provides clients of a proposed application with a physical representation of key parts of the web system before system implementation. Developers can use prototyping and iterative process development to discover the actual clients needs and their functional requirements at very early commencement of the project [7]. This paper proposes a prototyping web authoring toolkit that developer can use to create functional system with minimum understanding of the system and minimum amount of effort.

This paper will provide an overview of the web authoring toolkit for web system development. The toolkit integrates web technology with database system and includes a communication module that allows developers and clients to collaborate with each other through a chat services and other functions integrated in the web-based toolkit using Java technology. This paper will first introduce the proposed toolkit that can create functional system with minimum amount of time, and how this will improve the process in web application development. Next, the paper will discuss how

the communication tools improve the collaboration among developers and clients. After that, the paper will review the toolkit in an iterative development model and finally conclude with several advantages gained from using the web authoring toolkit.

## 2 Web Authoring Toolkit

The web authoring toolkit is designed with a three-layers architecture of database-driven Internet application. The objective is to provide the adaptability, extensibility and reusability of today business database structure. The design structure is to provide the flexibility of data medium and adaptability of the presentation interface. The physical database layer is the lowest layer in the architecture, following next the database structure layer and the highest the database connection layer (see figure 1). The physical layer is responsible to handle the traditional storage medium such as the RDBM database and the XML files. The database structure layer contains objects that are responsible for managing the table schema in the database. All high level queries are parsed and organized by the run-time objects in this layer. These queries are then broken down into simple standard SQL access or storage operation and passed to physical layer for execution. Lastly the database connection layer maintains the connection between the application program and the database system. This architecture allows the replacement of alternative storage medium in the physical layer with minimum of efforts. In addition, it allows flexibility and instant modification of the database scheme structure. This design breaks through the traditional fixed or static database schema and achieves the aim of flexibility in database structure. In the following, the paper will continue to explore these three different layers separately.



**Fig. 1.** Three-layers Design Architecture Structure.

2.1 Database Connection Layer

This layer provides the interface allowing the program accessing the database system. The *DB* (*\$szHost*, *\$szDatabase*, *\$szUser*, *\$szPassword*) is the constructor of the interface; it connects to the storage by providing the unique identification. The user can request the service by the *query* (*\$szQuery*, *\$QID=0*). The *next\_record* (*\$QID=0*) function can be used to retrieve the query result with the Query ID (*\$QID*). In addition, this function performs errors checking, prepares the current result row in the memory and increases the row counter for operation efficiency. The *current\_record* (*\$QID=0*) returns the current array of record in the memory, and the result will be released from memory when the *free* (*\$QID=0*) is executed.

2.2 Database Structure Layer

The database structure layer is the most important layer because its task is to organize the data structure in the database. Each original table structure is represented by myTable and myField (see Table 1 and Table 2). Both tables are identified by a unique ID: TID and FID respectively. The TID in myField table is a foreign key from the myTable table, which allows one table to contain multiple fields. The structure of these tables is used to represent the core information in project database schema. The combination of these tables achieves very high flexibility in the database structure. Each data structure is clearly defined, and allowed any changes in the myField table. These two tables are allocated in a separate database due to their frequent high access rates. Lastly, the myValue table (see Table 3) contains the actual data storage of all the data medium contents. All these values are referenced by the unique primary keys TID and FID, which have been defined in the myTable and myField tables.

Table 1. myTable structure

TID	Auto increment int
TableName	Text

Table 2. myField structure

FID	Auto increment int
FieldName	Text
FieldType	Text

Table 3. myValue structure

TID	Int
FID	Int
Value	Blob

For example, table 4 shows the data structure to store information for the entity ‘Student’. The attributes include the student number, student name and the age of the

student. The database structure layer re-represents the same data structure of the ‘Student’ in the myTable and myField tables (see Table 5 and Table 6). The unique TID and FID are the keys for accessing data values stored in the myValue table (see Table 7). This design separation resolves the dependency of data values on the data structure, and achieves a higher flexibility of dynamic data structure in the database system.

**Table 4.** Original student table values

StudentNo	Name	Age
1001	Alex	20
1002	Bob	25
1003	Cathy	30

**Table 5.** myTable value

TID	TableName
1	Student

**Table 6.** myField value

StudentNo	FieldName	Type
1	StudentNo	Text
2	Name	Text
3	Age	Int

**Table 7.** myValue value

TID	FID	Value
1	1	1001
1	2	Alex
1	3	20
1	1	1002
1	2	Bob
1	3	25
1	1	1003
1	2	Cathy
1	3	30

From traditional web system development, changes in the data structure will likely involve tedious of works and maximum amount of time in order to make the necessary modification from old data structure to the new data structure. It involves the changes in the database schema and the actual codes in every database-driven web page. This might actually involve many complications for just a very small change in the schema. For example, if the previous ‘Student’ data structure requires an extra attribute ‘address’, most developers will have to manually change the schema in database and modify every page containing the ‘Student’ entity. However, with the prototyping web development approach, developers only require to make the changes in

the myField table through the user-friendly web authoring toolkit without any further complications. This design approach overcomes the problem in contents changing due to client uncertainty at the commencement of project, and eventually provides a flexible and dynamic data structure in the web prototyping development process.

### **2.3 Physical Database Layer**

Physical database layer provides an adaptive interface between the database structure layer and number of different storage mediums. It provides the ability for accessing different storage medium. For each storage medium, it has its own way for establishing the connection and own format of querying statement. To allow developers switching between different storage mediums, this layer is designed to provide such ability with minimum of modifications of several functions in this layer. These are the core functions for accessing the storage mediums, and they include the functions for establishing a proper connection to storage mediums and the query or execution statement for manipulating the data in the medium. With the current approach, the database connection layer and database structure layer can remain unchanged, and increase the compatibility and extensibility of the entire model.

## **3 Collaboration in Web-Based Toolkit**

The web authoring toolkit is designed with a very comprehensive set of communication component on top of the core web authoring toolkit framework. This communication module is designed and implemented according to the concept of CSCW (Computer Supported Cooperative Work) [8]. Individually, the communication tools that the toolkit is designed of have many competitors that are currently in use by the corporate community. The collaboration component includes the following features sub modules:

### **3.1 E-Chat Module**

The E-Chat module is written entirely with core Java and Java applet. The applet had to be compatible with Java 1.1.5 so as to work with Microsoft's default Java VM plug-in for Internet Explorer. The objective of this design is to ensure that the applet can be run from the most common web browser (i.e. Internet Explorer or Netscape Communicator). The online chat system is designed to be easily extendible and configurable no matter what sort of system it is deployed on, with minimal dependencies. The server includes a multithreaded object class, which spawns threads when clients connect through the Java applet embedded in the dynamic html page. A maximum client limit is imposed so as to hinder Denial of Service attacks. This design enables the server to handle multiple clients without heavy demand on the system.

The client's Java applet is designed to run from any web browsers. It is designed in a highly object-oriented behaviour, with the separation between the design patterns

in the back-end functions and the front-end graphical user interface. In addition, the client and server communicate via TCP socket, which provides reliability and error correction in the network environment. Every session is saved in the database system. The users may, at a later stage, view the session information and the chat contents in each session. Developers and clients can review any previous online chat discussion at any time. In the chat module, the system also provides a 'File Transfer' function. This function allows active user in chat session to send file to other user in the same session. In most existing system, users can only exchange files through email. Thus, any user without email is not able to communicate with other users. This again shows that this toolkit has provided another alternative way for developers and clients to communicate in a much efficient way.

### 3.2 Electronic Whiteboard

In order to further improve the collaboration for developers and clients in their development process environments, the idea of having the electronic whiteboard is implemented in the web authoring toolkit. The whiteboard allows developers and clients to perform drawing on the Internet. They can share ideas, communicate in a graphic ways. They can enter text; draw lines, rectangles, scribbling and ovals of any color. The whiteboard is designed to have the option of saving the drawing and opening previous saved drawing or other standard image files. Again, this greatly improves the communication and collaborative group work between the users. This component is similar to the E-Chat module, it is implemented purely in Java, and thus users can operate from the standard web browser on any platform.

### 3.3 Synchronized Group Web Browsing

Synchronized group web browsing is the next feature communication tool in the toolkit. With the combination of chat and synchronized web browsing, users can be notified automatically when anyone joins or leaves the session. The system is just like a virtual discussion room and the users in the same session can interact with each other. In addition, developers or clients can direct active users' browser to designated page by setting the URL address. Unlike other solutions, by utilizing Java technology this module can be directly integrated with the webs without any additional effort. Figure 2 shows the interaction between users' browsers in the virtual discussion room.

Developer 'Jesse' creates a new session; client 'Chee' joined the developer in the chat session. The developer can set a URL address in the chat applet. When a URL is set, every developer and client in same session will have browsers displaying the contents from the specific URL. Developer can use this tool to demonstrate the functional system on the Internet. Clients can provide online feedback to the developers regard the system. This achieves the basic requirements of a virtual collaboration work place. However, not many online system supports such feature. The toolkit is designed to be one of the few web systems that offer this innovative function.



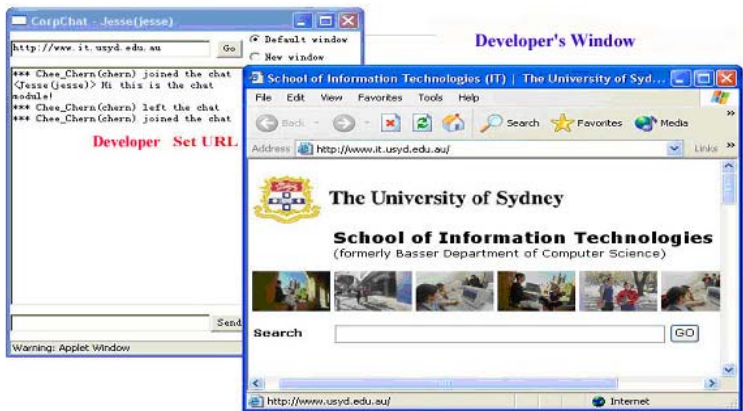


Fig. 2. Developer set URL on the applet.

4 Toolkit Benefits in an Iterative Web Development

Software engineering researchers introduce an iterative design model (see figure 3) as they realized that software development life cycle phases appear with significant overlap in web development. These overlaps are caused by clients having difficulty in articulating the system requirements at the commencement of the project.

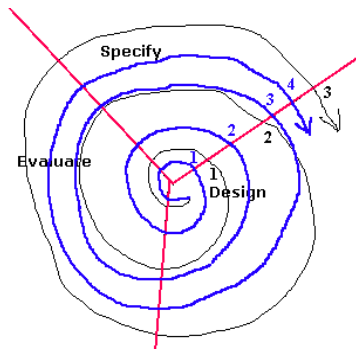
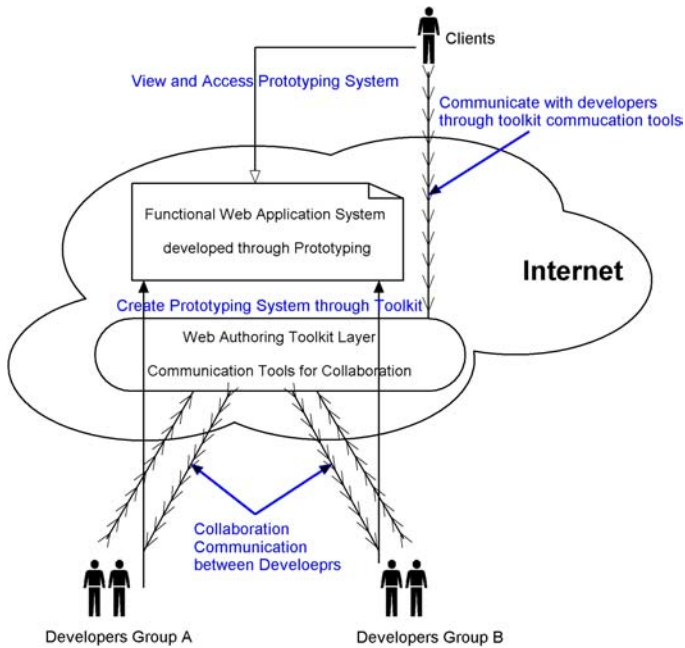


Fig. 3. An Iterative Design Model Comparison.

In figure 3, the black line represent the time involved in a web development, and the blue line represent the time involved in the same web development with the assistance of the web authoring toolkit. It shows that blue line has more number of phases overlap than the black line in the same time frame. This means that the blue line indicates a faster development process. This result is expected since the prototyping web authoring toolkit allows developer to create functional system with minimum understanding of the system and minimum amount of effort. With the prototyping functional system, clients will be able to understand their needs sooner and provide early

feedbacks to the developers. Consequently, less changes of the system will likely to occur and the final actual system will be most likely to be completed on time with estimated budget and without additional hidden cost.



**Fig. 4.** Interaction Between Developers and Clients.

Figure 4 shows the collaboration between the developers and clients in the toolkit environment. Through the toolkit, developers from group A and group B can develop the system concurrently without waiting the tasks completion of neither group such as functional modules belongs to database department or the completion of template layout belongs to graphic design department. Developers can work individually on separate section without interfering each other's works. In addition, developers can still communicate to each other though the sophisticated communication tools. This applied to clients as well where clients can provide online instant or offline feedbacks to the developers. This not only improves the collaboration work among developers themselves, but also improves the communication between the developers and the clients.

## 5 Conclusion

This paper introduces a web authoring toolkit for rapid prototype web development. The unique characteristics of web development have become the attentions of many software engineering researchers. Researchers found that often the client will have

difficulty articulating the system requirement at the commencement of web system project. Clients are only able to express their needs in term of solutions at the beginning of project, and they are able to formulate the detailed requirements as the system is being developed. Consequently, the frequency of client's requirement changes had a surprisingly high correlation. This results a major issue as the hidden cost in web application projects are increased significantly. The web authoring toolkit is proposed to resolve the problem. The web authoring toolkit provides a fast track design route for new web system to be designed and assessed by clients by providing a minimum amount of efforts. Developers can create functional web system through rapid prototyping development. In addition, the toolkit is embedded with a set of communication tools. The communication component is designed according to the concept of CSCW, and it improves the collaboration works between developers and clients. Finally, the web-authoring tool itself is a web-based application, which is platform independence and interoperability, and can be installed on any platform of operating system.

## References

1. Christian W.: Portal Server Technology. IEEE Internet Computing (2002)
2. England E., Finney A.: Managing Multimedia: Project Management for Interactive Media. 2nd edn. Addison-Wesley (1999)
3. Overmyer S.: "What's different about requirements engineering for web sites?". Requirements Engineering Journal. Vol. 5, No. 1 (2000) 62-65
4. Russell P. "Infrastructure – make or break your e-business", in TOOLS-Pacific 2000: Technology of Object-Oriented Languages and Systems, Sydney, Australia, 2000
5. David L., Brian S.: Characteristics of Web Development Processes. SSGRR-2001: Infrastructure for e-business, e-education, and e-science.
6. David L., John. E: <http://www2002.org/CDROM/alternate/678/>
7. Stacey D. Lectures on Rapid Software Prototyping. University of Guelph, Ontario, Canada. (1997). <http://hebb.cis.uoguelph.ca/~dave/343/Lectures/prototype.html>
8. Bentley, R. & Horstmann, T. & Trevor, J.: 'The World Wide Web as enabling technology fro CSCW: The case of BSCW', Computer Supported Cooperative Work: Special issue on CSCW and the Web Vol. 6. (1997)
9. WebCon: design and modeling of database driven hypertext applications Sommer, U.; Zoller, P.; System Sciences, 1999. HICSS- 32. Proceedings of the 32nd Annual Hawaii International Conference on , Volume: Track6 , 5-8 Jan. 1999 Pages:8 pp

# Capacity Planning for Composite Web Services Using Queueing Network-Based Models\*

Dunlu Peng<sup>1,2</sup>, Yang Yuan<sup>1</sup>, Kun Yue<sup>1</sup>, Xiaoling Wang<sup>1</sup>, and Aoying Zhou<sup>1</sup>

<sup>1</sup> Department of Computer Science and Engineering  
Fudan University, China

{dlpeng, yyuan, kuny, wxling, ayzhou}@fudan.edu.cn

<sup>2</sup> College of Computer Engineering  
University of Shanghai for Science and Technology, China

**Abstract.** In this paper, queueing network-based models are proposed to predict the performance of composite Web services and make them run in an optimal way with limited resources. The models are necessary constituents of reliable Web service composition. They are flexible enough to evaluate the performance indices of the composite Web services simply by solving some equations rather than doing a lot of exhausting experiments. Research shows that our models provide an effective way for both running the services optimally with certain resources and estimating the growth needs accurately.

## 1 Background

With the boom in the new online economy, more and more e-commerce companies put their core business on the Internet in the form of Web services [16]. From a business-to-business integration (B2B-integration) point of view, the idea that stands behind the Web services paradigm is to make business over the Internet more effective and reliable [8]. In order for this idea to become reality, certain standards have been developed, such as SOAP[w3.org/TR/soap], UDDI[uddi.org], and WSDL[w3.org/TR/wsdl]. Besides enabling Web services to realize the idea of effective B2B-integration, these technologies promise an ocean of Web services that can be invoked remotely using standard XML-based protocols.

As the number of Web services grows, the competition among the companies becomes drastically more and more. In order to survive the massive competition, the providers must make their services more attractive with high qualities. Some qualities, such as low cost, speed of delivery and so on, determined by managing people are out of the scope of technologies. Others, such as less response time, less errors and higher availabilities, must owe to technology organizations. If a service does not have sufficient capacity to serve all of its requestors, it will frustrate the requestors with slow response time, timeouts, errors and broken links. As a result, the requestors may choose to go elsewhere to find other Web services which have better service, higher

---

\* This work is supported by the National Natural Science Foundation of China under Grant No. 60228006, and the National Hi-tech R&D Program of China under Grant No. 2002AA116020.

quality, and faster speed. In e-commerce applications, such service behavior could translate to sizable revenue losses.

The key to the problems mentioned above is capacity planning for the Web services before they impact on the business. In other words, an accurate and effective way must be found to evaluate the performance before you invest the Web services. Research on this area has been limited compared to other issues on Web services such as publication, discovery and composition [1, 8, 16]. In this paper, queueing network-based model are proposed for composite Web service capacity planning. With the models, we can measure the performance indices simply by solving some equations which avoid doing exhausting experiments and simulations. The factors influencing the performance of the composite Web service are also analyzed. Experiments are conducted to show the accuracy and efficiency of the model.

The paper is organized as follows: In Section 2, we will introduce capacity planning for Web services and queueing network-based modeling briefly. The design and validation of queueing network-based models for composite Web service are described in Section 3. Section 4 presents one computational algorithm for the models. In Section 5, we experiment with a case of composite Web Service and a limited resource to show the accuracy and efficiency of our models. Related works are discussed in Section 6. Finally, we draw our conclusions in Section 7.

## 2 Capacity Planning for Web Services and Queueing Network Modeling

### 2.1 Capacity Planning for Web Services

Capacity planning for Web services is the process of measuring the services' ability to serve contents to their requestors in an acceptable time. In other words, Web service capacity planning is the process of monitoring and projecting service workload and specifying the most effective computing environment to meet future demands given a small number of parameters [15]. Given infinite resources, the expected quality of service can always be provided, but with limited resources, capacity planning and resource management is very useful.

There are multiple ways to measure the performance of a system. The most commonly used performance metrics are *response time* ( $R$ ) and *throughput* ( $X$ ) [2]. To a Web service, the *response time* is defined as the time interval from a request arriving at the service to the instant the corresponding reply begins to appear at the requestor's terminal, and it is determined by two factors: the quality of network transmission, and the processing capacity of the service. In this paper, we only consider the processing capacity of the service. The quality of network is not within the scope of this paper and being extensively studied in literature [3]. The *throughput* is generally considered as a measure of the service's productivity, that is, the number of requests served successfully during the measurement period.

Besides response time and throughput, some other metrics for the performance of Web service are also considered in our models, such as utilization, service rate, the length of request queue and so on. Their definitions and influence will be discussed in section 3.

## 2.2 Queueing Network Modeling

Queueing network modeling [15] is a general and widely used analytical modeling technique for system performance evaluation. It has been extensively applied to represent and analyze resource sharing systems, such as production, communication and computer systems. A queueing network model is a collection of *service centers* representing the system resources that provide service to a collection of *customers* that represent the users. The analysis of the queueing network models consists of evaluating a set of performance measures, such as resource utilization•throughput and customer response time.

Queueing network models are sets of mathematical formulae that are devised to capture the relationships between workload and performance measures. With specific parameter values, it is possible to evaluate performance measures of interest by solving some simple equations. This model can easily be developed since its parameters are based on directly measurable quantities, such as the arrival rate, service rate and etc. For more elaborate introduction to queueing network modeling, the reader is referred to [2,4,5,12,14,15].

The use of queueing network models for evaluating the performance of composite Web services is justified by many reasons. For example, it is straightforward to map the request behavior of a Web Service into a queueing network. *Web services* are modeled by service centers and *the requestors* are modeled by customers. Another important reason is that queueing network models have a good balance between a relative high accuracy in the performance results and the efficiency in model analysis and evaluation [15].

## 3 Queueing Network-Based Models for Composite Web Services

### 3.1 Web Services Composition

Web service composition refers to the process of creating customized services from existing services by a process of dynamic discovery, integration and execution of those services in a deliberate order to satisfy user requirements [7].

The formalization of relationships between sub-services in a composite Web service has been proposed in [16] and [8] with different methods. We summarize some of them as follows: *sequence* relationship ( $S_i \rightarrow S_j$ ) means a composite service that performs the service  $S_i$  followed by the service  $S_j$ , *unordered sequence* relationship ( $S_i + S_j$ ) means a composite service that performs either the service  $S_i$  followed by the service  $S_j$  or  $S_j$  followed by  $S_i$ , *alternative* relationship ( $S_i // S_j$ ) means a composite service that behaves as either service  $S_i$  or service  $S_j$ , and iteration  $\eta S_i$  represents a service that performs a certain number of times the service  $S_i$ . For some other kinds of relationship, the reader can see [16,8].

### 3.2 Queueing Network-Based Models for Composite Web Services

Before modeling, we assume that all the services have infinite buffer for waiting requests and serve the requests with *first-come-first-serve* (FCFS) model [14], which is

extensively applied in Web servers and application servers. Another assumption is, for a provider, each Web service only supported on one server.

First simple queueing system has been proposed to model a single Web service, and then we model the composite Web services using a queueing network model by extending the queueing systems we built for the single Web services.

### 3.2.1 Queueing Systems for a Single Web Service

A single Web service model is described by an arrival process of incoming requests, a service process, a buffer for holding the waiting requests and a scheduling algorithm of the queue, shown in figure 1 (a). We can simplified the model as arrival process, the waiting requests and service process by the assumptions given at the beginning of section 3.2 .

**Definition 1 (Service Rate)** Given a time interval  $\Delta T$ , and the number of requests completed successfully at Web service  $S_i$  within  $\Delta T$  is  $c$ , then the service rate  $\mu_i$  is  $c/\Delta T$ . Sometimes the service rate is also called *throughput* or *departure rate*.

**Definition 2 (Queue Length)** At certain time, the number of requests waiting in the queue is  $w$  and being on served is  $r$ , then the queue length of service  $S_i$  is  $w+r$ , denoted as  $Q_i=w+r$ .

**Definition 3 (Arrival Rate)** During the measuring period  $\Delta T$ , If the number of requests joining in the queue of service  $S_i$  is  $a$ , then the arrival rate  $\lambda_i$  is  $a/\Delta T$ .

**Definition 4 (Stable Service)** A service is stable when its service rate  $\mu$  is not less than the arrival rate  $\lambda$ , that is  $\mu \geq \lambda$ .

Given a stable Web service with request arrival rate  $\lambda$  and service rate  $\mu$ , by the queueing theory described in [5] we can get the probability of  $k$  requests as flows:

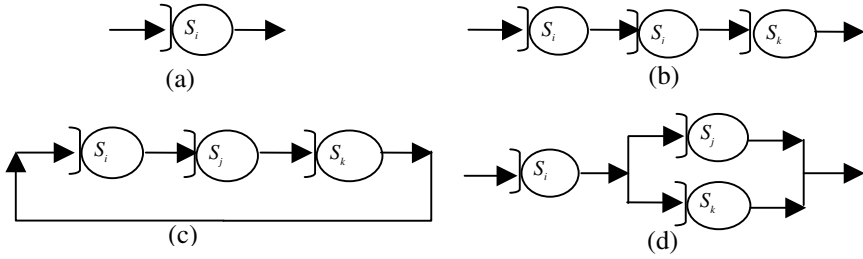
$$\pi(k) = \rho^k (1 - \rho) \quad k \geq 0 \quad (1)$$

Where  $\rho = \lambda/\mu$ . The average queue length is  $\tilde{Q} = \rho / (1 - \rho)$ , and the average response time is  $\tilde{R} = 1 / (\mu - \lambda)$ .

Function (1) gives the mathematic model for evaluating the performance indices for a single service. By extending the model, we can immediately model a composite Web service as a queueing network which is composed with several single queueing systems with the relationship we introduced above.

### 3.2.2 Queueing Network-Based Models for Composite Web Services

For a composite Web service, the queueing network model can be seen as some interconnected queueing systems for single Web services. The interconnections between sub-services form the topology of the queueing network. The topology of a queueing network shows the relationships between the services and the movement of the requests among them. In our models, we consider the topology for the Web services having relationship shown in figure 1. (b) sequence relationship, (c) iteration of the services and (d) compound network topology for the composite Web services.



**Fig. 1.** Topologies of queueing network for composite Web service. The arrows denote the moving direction of requests. (a) the single Web service; (b) the sequence Web services; (c) iteration of Web services; (d) compound relationship

### 3.2.3 Mathematic Models for Composite Web Service Queueing Network

Queueing network analysis is based on the definition and analysis of an underlying stochastic process that is usually a discrete space continuous time homogeneous Markov process [5]. In the composite service queueing network, a state includes a Web service and the number of requests in the queue. Consider a network with  $M$  services, let  $Q_i$  denote the number of requests in the queue at service  $i$ ,  $Q = \{Q_1, Q_2, \dots, Q_M\}$  is the queue joint length and  $\pi(Q) = \pi\{Q_1, Q_2, \dots, Q_M\}$  is the queue joint length distribution. Let  $\pi$  denote the state probability vector of the Markov process and  $P$  its transition rate matrix is defined by the normalized solution of the following linear system:

$$\pi P = 0 \quad (2)$$

which is also called system of *global balance equations* [5]. Performance indices of the queueing network can be derived by the state distribution of the process.

The joint queue length probability  $\pi$  defined by the solution of the associated Markov process, given by the linear system (2) has a *product form solution* [4], as follows:

$$\pi(Q) = \frac{1}{K} V(Q) \prod_{i=1}^M f_i(Q_i) \quad (3)$$

where  $K$  is a normalizing constant,  $Q$  is the total network population, function  $V$  is defined in terms of network parameters and  $f_i$  is a function of state  $Q_i$  and depends on the type of service  $i$ ,  $1 \leq i \leq M$ .

Equations (2) and (3) can be analyzed by efficient algorithms with a polynomial time computational complexity in the number of network components. This class of models allows a good balance between a relative high accuracy in the performance results and the efficiency in model analysis and evaluation [15].

## 4 Computational Algorithms

The main advantage of the models we built is product form queueing network which has several efficient algorithms been developed for their performance analysis [5]. Here we introduce a popular algorithm, Convolution Algorithm, which was proposed in [14].



For a queueing networks the computation of the state distribution  $\pi$  requires the evaluation of the normalizing constant  $K$  in formula (3). Since  $V(Q) = 1$ , constant  $K$  can be defined as  $K = \sum_n \prod_{i=1}^M f_i(Q_i)$ . Direct computation of  $K$  as a summation over all the feasible states  $Q$  of the network would take an exponential time in the number of services and request of the network. The Convolution Algorithm avoids this direct computation and evaluates  $K$  recursively. For a network with  $M$  services and  $N$  requests, let  $K_j(n)$  denote the normalizing constant of the network with  $n$  requests and the first  $j$  services,  $1 \leq j \leq M$  and  $0 \leq n \leq N$ . Then  $K = K_M(N)$  and we can write the recursive relation that is the convolution

$$K_j(n) = \sum_{i=1}^{j-1} f_i(Q) K_{j-1}(n - Q_i) \quad (4)$$

where  $K_j(0) = 1$ . According to the assumption shown at the beginning of section 3.2, we know that sub-service  $j$  is only supported by one server, thus we simply have  $f_j(Q) = \rho_j^n$ . Hence convolution (4) reduces to

$$K_j(n) = K_{j-1}(n) + \rho_j^n K_{j-1}(n - 1) \quad (5)$$

Therefore the time computational complexity of evaluating  $K = K_M(N)$  is  $O(MN)$  operations. It is worthwhile noticing that several performance measures can be directly evaluated by function  $K_M$ . For a single provider service  $j$  with arrival independent service rate, we can write:

Queue length distribution

$$\pi_j(n) = \rho_j^n [K_M(Q - n) - \rho_j K_M(Q - n - 1)] / K_M(Q) \quad (6.1)$$

Average queue length

$$\tilde{Q}_j = \sum_{n=1}^Q \rho_j^n K_M(Q - n) / K_M(Q) \quad (6.2)$$

Throughput

$$X_j = \lambda_j K_M(Q - 1) / K_M(Q) \quad (6.3)$$

Utilization

$$U_j = X_j / \mu_j \quad (6.4)$$

However, for a sub-service  $j$  with arrival dependent service rate the queue length distribution can be written as following  $\pi_j(n) = f_j(n) K_{M-\{j\}}(Q - n) / K_M(Q)$ , where  $K_{M-\{j\}}$  is the normalizing constant of the entire network except for node  $j$ .

## 5 Experiments

From the discussion above, we know that using the queueing network models to evaluate the performance metrics is just to solve a set of equations. We can assign the resources in an optimal way if the computed metrics are accurate. To verify the flexi-

bility of the models, we performed some experiments to compare the performance of the resource assignment guarded by the optimal results of our models with a random method.

	Entry	Al	Ht	Cr	Bf	Departure
Entry	0	43	45	8	4	0
Al	0	5	28	32	5	30
Ht	0	16	5	39	10	30
Cr	0	10	9	6	26	49
Bf	0	5	6	10	10	70
Departure	0	0	0	0	0	0

Fig. 2. Transition probability matrix of our example

We use the example of business trip service which is also appeared in [16]. The service contains four sub-services: Airline (Al), Hotel (Ht), Car rental (Cr) and Breakfast (Bf). Requests move between sub-services in a composite Web service can be presented as state machine [9]. Figure 2 shows the transition probability (in percentages) matrix among the stage vector services: Al, Ht, Cr and Bf. The row vectors show the transition probability from one stage to another. For example, the second row indicates that the transition probabilities from Al to the entry, Al, Ht, Cr, Bf and leave the composite service are 0%, 5%, 28%, 32%, 5% and 30%, respectively. The transition probability shows the possible relationship between sub-services in a composite Web service.

5.1 Experimental Setup

The computers we used in our experiments contain four different servers acting as the service providers and several clients acting as service requestors. The four servers, numbered as 1#, 2#, 3# and 4#, are different in the processing abilities which are 120, 60, 60 and 120 and weighted by the configurations, such as CPU, memory and other devices. The clients used in the experiments are several PCs having same configurations. The way a request picks a service is depended on the probabilities given in figure 2.

Firstly, we made an assignment of the servers to the Web services in a random way. For example, we assign Al to server 1#, Ht to server 2#, Cr to server 3#, and Bf to server 4#, simply presented as {1#.Al, 2#.Ht, 3#.Cr,4#.Bf}, where x#.Y means allocating server x# to the service Y. In figure 3, 4, 5, it is called *initial state*.

Secondly, we chose the way of allocating resources to the sub-services with the least overall average response time by solving the queueing network models. In our example, it is observed that {1#.Cr, 2#.Bf, 3#.Al, 4#.Ht} is the optimal policy for resource allocating because it has the least overall average response time. So we chose it for comparison. In figure 3,4,5, it is called *improved state*.

For better comparison, we preferred the performance parameters which are varied significantly, such as the overall average response time and throughput of the composite service, as the primary performance metrics for our analysis. We also analyzed the distribution of queue length for each sub-service.

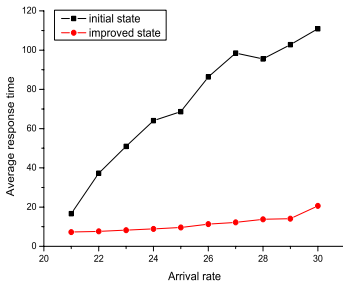


Fig. 3. Average response time comparison

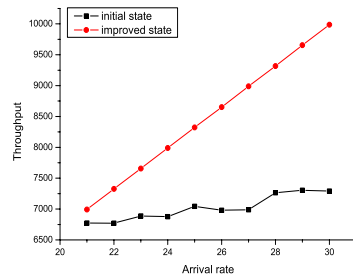


Fig. 4. Throughput comparison

## 5.2 Experimental Results

It is observed from figure 3 that, the optimal result of queueing network model can significantly improve the composite service performance by reducing the overall response time. As the arrival rate goes up, the overall average response time of the improved state also varies for different arrival rate but is much shorter than those of the initial state. We presented the throughput of both assignments in figure 4, from which we can see that the throughput of the improved state rises rapidly as arrival rate increases, this situation is same to Figure 6 which depicts the overall average response time comparison.

Figure 5 presents the distribution of the queue length at each sub-service under the two assignments as the request rate is at 21, 22, 23. >From the figures, we observed that in the initial state (on the left side of the figure), the distribution of queue length is quite unbalanced. Especially, at Cr and Ht, the queue length is much longer than that at other services, which means that in order to be served the requests must wait for a long time, This is due to the servers' abilities are insufficient to provide the service for its requests. By contraries, at Al and Bf, the length of queue is very short, which means the request served with little waiting. In other words, the server's ability is excessive to serve the requests. In the improved state (on the right side of figure), the length of queue distributes quite fair and is not too long. It is because each service is competent for processing its workload.

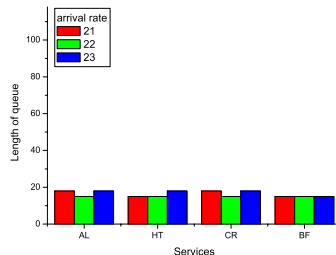
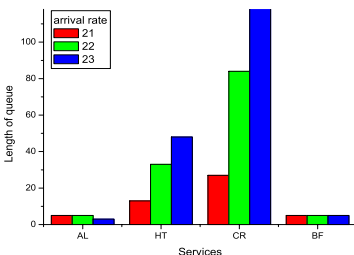


Fig. 5. Comparison of request queue length distribution. The left side is initial state and the right side is for improved state.

## 6 Related Work

Although excessive of work on Web service have addressed the composition issues in Web services [7,8,9,10,16], the studies on the performance has been limited. Study of quality of service (QoS) in network transmission is active in recent years, including efforts of building Integrated Services (IntServ) architecture with end-to-end QoS guarantee, and Differentiated Service (DiffServ) architecture with alternative levels of services provisioning [3]. Bhatti and Friedrich addressed the importance of server QoS mechanisms to support tiered service levels [11]. In context of capacity planning, [12] provides a model based on bandwidth demands for memory, processors data bus and I/O buses. However, network layer QoS is not sufficient in support any Web services, especially, the composite Web services. Our research, capacity planning for Web services, can be seen as a complement of that area.

## 7 Conclusion

Capacity planning is an important aspect of developing and deploying high quality Web services. Proper planning ensures healthy Web services that can grow to meet future needs. In this paper, we proposed queueing network-based models for both single Web services and composite Web services. By the models, we can evaluate performance measures of interest by solving some simple equations. Experiments show that the obtained measures help in accurately predicting the performance of composite Web services under certain assigned resources and workloads. This is essential for understanding the limitations of composite Web service. With this information, one can assess the capacity of the services, maximize its performance and arrive at an optimization between the Web services and the resources. In our future work, we will research on scheduling services under certain workload considering the relationship between them.

## References

1. Hoschek, W.: The Web Service Discovery Architecture. Proceedings of the 2002 ACM/IEEE conference on Supercomputing, 2002, Baltimore, Maryland.
2. Lam, S., Chan, K. H.: Computer Capacity Planning: Theory and Practice. Academic Press, 1987.
3. Internet Engineering Task Force, <http://www.ietf.org>
4. Balsamo, S.: Product Form Queueing Network, Performance Evaluation , P377-401,2000.
5. Buzen, J. P.: Queueing Network Models of Multiprogramming. Ph.D. Thesis, Harvard University, Cambridge, MA, 1971.
6. Wodtke, D., Weikum, G.: A formal foundation for distributed workflow execution based on state charts, In Proceedings of the 6<sup>th</sup> distributed International Conference on Database Theory, Delphi, Greece,1997.
7. D.Chakraborty, F.Perich, A.Joshi, T.Finin and Y.Yesha. A Reactive Service Composition Architecture for Pervasive Computing Environments. In 7th Personal Wireless Communications Conference (PWC 2002), Singapore.

8. Limthanmaphon, B., Zhang, Y.: Web Service Composition with Case-Based Reasoning. In Proc. of the Fourteenth Australasian Database Conference (ADC'03), Feb. 2003, Adelaide, Australia, Australian Computer Society.
9. Banatallah, B., Dumas, M., Fauvet, M.C., Paik, H.Y.: Self-Coordinate and Self-Traced Composite Services with Dynamic Provider Selection. Technical Report, UNSW-CSE-0108, 2001.
10. Banatallah, B., Dumas, M., Fauvet, M.C., Rabhi, F.A.: Towards Patterns of Web Services Composition. Technique report, UNSE-CSE-TR-01111, 2001.
11. Bhatti, N., Friedrich, R.: Web Server support for tiered services. *IEEE Network*, pages, 64-71, September/ October, 1999.
12. Kant, K., Won, Y.: Server Capacity planning for Web Traffic Workload. In *IEEE Transactions on Knowledge and Data Engineering*. Vol, 11, No.5, September, 1999.
13. Chen, X., Mohapatra, P., Chen, H.: An Admission Control Scheme for Predictable Server Response Time for Web Accesses. In proceedings of 10<sup>th</sup> WWW Conference, Hong Kong, May 2001.
14. Buzen, J. P.: Computational algorithms for closed queueing networks with exponential servers" *Comm. of the ACM*, Vol. 16, No. 9, pp. 527-531, 1973.
15. Lazowska, E., Zahorjan, J., Graham, S., Sevcik, K.: *Quantitative System Performance: Computer System Analysis Using Queueing Network Models*, Prentice Hall, Englewood Cliffs, N. J., 1984.
16. Hamadi, R., Benatallah, B.: A Petri Net-based Model for Web Service Composition. In *Proceedings of the Fourteenth Australasian Database Conference (ADC'03)*, Feb. 2003, Adelaide, Australia, Australian Computer Society.

# A Web-Based Coordination Infrastructure for Grid Collective Services\*

Javier Jaén<sup>1</sup>, Jose H. Canós<sup>1</sup>, and Elena Navarro<sup>2</sup>

<sup>1</sup> Department of Information Systems and Computation  
Polytechnic University of Valencia  
Camino de Vera s/n, Valencia, Spain  
{fjaen,jhcanos}@dsic.upv.es

<sup>2</sup> Computer Science Department  
University of Castilla-La Mancha  
Avda. España S/N, Albacete, Spain  
enavarro@info-ab.uclm.es

**Abstract.** Virtual Organizations (VO) consisting of heterogeneous institutions and individuals that share resources dynamically and in a coordinated way to support collaborative problem-solving are emerging in many fields. Consequently, new types of distributed infrastructures known as Grids have been proposed to cope with these new sharing requirements. Grids offer sets of collective services to support collaborative tasks which are distributed in nature and need asynchronous communication. Existing approaches to this problem lack of flexibility, adaptability and are tightly bound to the collective service that is provided. We present here a generic event model to build collective services that requires no programming, is extensible and can accommodate different event vocabularies. An implementation is also described using standard Web technologies like Java Servlets, XML, XSLT, and XSQL.

## 1 Introduction

According to Pfister [1] there are in general three ways to improve performance: (i) work harder, (ii) work smarter, and (iii) get help. In terms of computing technologies, this means respectively (i) using faster hardware, (iii) using more efficient algorithms, and (iii) interconnecting processor and memory systems. Grids [3] are new emerging massive heterogeneous distributed infrastructures which implement the third approach. They provide support for the management and exploitation of Virtual Organizations: communities which differ on size, purpose of collaboration, types of resources being shared, and security policies. But all of them have a fundamental common characteristic: they are all composed by participants that do not necessarily trust each other yet want to share resources to fulfill a common goal.

The architectural components of Grids, their purpose, and interaction models are defined in what is known as the Grid Architecture [4]. In this architecture, several layers deal with the problem of how to interact in a uniform way with any type of shared

---

\* This work has been partially funded by the Spanish CICYT project DYNAMICA - **DY**-**N**amic and **A**spect-Oriented **M**odeling for **I**ntegrated **C**omponent-based **A**rchitectures (TIC2003-07776-C02-02).

resource. A defined set of collective services, such as *Directory services* [5], *Data replication services* [6] or *Community authorization services* [7] (see [4] for a detailed list of collective services), take care of interactions across collections of these resources. Experience shows that such collective services do not scale well if implemented as centralized services [5, 8]. Moreover, if the volume of data to be supported by such systems is not negligible then performance may be severely affected. This means that with a distributed design in mind coordination requirements arise to allow: first, a distributed data repository infrastructure, and second, a loosely-coupled asynchronous communication. This is exactly the model supported by event systems [2].

Some illustrating examples where coordination is needed and events model could collaborate are: a distributed replica manager keeping consistency among all catalogs and coordinating with data movers to perform the most efficient data transfers or a monitoring system requiring notifications when exceptions occur on grid resources (producers) forwarding them to all interested consumers. Moreover, coordination becomes even more complex when trying to integrate core collective services in order to build sophisticated grid applications.

The previous examples impose several requirements that must be taken into account when defining a suitable coordination model for collective services:

- *Flexible information exchange*: across different collective services by passing either the entire dataset, or perhaps filtered content to the participating services. Besides, flexibility also means possibility to define domain-specific vocabularies.
- *Integration of data and event models*: without losing flexibility. Most collective services are massive data repositories that need to perform asynchronous notifications to other parties when data manipulation operations occur.
- *Security*: real production systems must implement security features so that malicious clients cannot execute attacks against a service or access private information.
- *Adaptability*: coordination models are usually implemented at a very low level (usually embedded within the middleware. For instance, if one would like to implement a new type of coordination model between Grid Index Information Services (GIIS) and Grid Resource Information Services (GRIS) [5] a new version of the middleware would be required which does not facilitate deployment and management in large-scale environments.
- *Architectural Flexibility*: collective services adopt specific architectural layouts to cope with their specific distribution requirements. Any distributed event layer that would impose any type of architectural restriction on how components need to be interconnected would not be appropriate.

We will show in the rest of the paper that we can satisfy all these requirements using standard web technologies like XML, XSL transformations, X509 certificates and java servlets running on standard web servers. Section 2 defines a distributed event model in the context of this technology. Section 3 discusses the main differences our approach with respect to other works and, finally, section 4 presents our conclusions and future research.

## 2 A Distributed Event Model for Xsql Pages

Performance and interoperability are key factors when thinking about technologies for data storage and exchange for collective services. Fortunately, existing and standard

database technologies, such as SQL, XML and HTTP cope already with these requirements. There are a wide number of products that address the integration of all three SQL, XML and HTTP. Among them, it is particularly interesting for us XSQL [10] for several reasons. First, it automates the use of underlying XML technology components to interact with relational repositories on the web; second, it can be extended in a very flexible way to define new behaviors; and third, because it is already in use in several Grid projects, such as European Datagrid [9], to prototype access to data repositories of some collective services.

## 2.1 Event Data Model

Unlike some XML event descriptions defined in the context of a particular application domain (e.g. event descriptions for Grid monitoring applications [11]), our approach is generic and does not define a domain specific representation for events. We propose an XML data model for events that can be extended with domain dependent vocabularies to deal with specific information requirements of a given application.

In our model, an event has a name, a timestamp signaling the moment in which the event was produced at its source, and routing information to trace the transformations that have been applied to the initial event. This information constitutes the header of a primitive event. Additionally, an event has a body containing an XML fragment which results from some XSL transformation. An example of event obtained applying an XSL transformation to the XML result obtained in the previous section is:

```
<?xml version="1.0"?>
<EVENT Name="CoordinateReadEvent">
  <HEADER>
    <TIMESTAMP></TIMESTAMP>
    <ROUTING>
    <NODE -
url="http://yourmachine.com/xsql/demo/point.xsql?x-
coord=11" transform="single-point.xsl"/>
    </ROUTING>
  </HEADER>
  <BODY>
    <POINT> <X>11</X> <Y>17</Y> </POINT>
  </BODY>
</EVENT>
```

Note that nothing prevents from using an alternative XSL transformation to define, for instance, a different type of event called PolyLineReadEvent whose body contains all or a subset of the coordinates produced by the query shown in the previous section. It is also important to see that we get for free all the expressiveness of XSL to transform any XML fragment produced by any arbitrary XSQL tag (including the user-defined ones) to produce any type of event structures. The combination of XSL flexibility plus XSQL extensibility results in an endless number of possibilities to define specific domain event vocabularies. To do so, each domain just needs to provide its collection of XSL pages that define the representation for the events that are relevant within the domain.

To implement this model within XSQL we need to extend the language with a new action element `<xsql:event-set>`. This element will delimit the XML fragment that is



to be transformed into events according to some explicit XSL transformation. Our XSQL page would look now as follows:

```
<?xml version="1.0"?>
<xsql:event-set transform="single-point.xsl",
    post="http://a.subscription.server/EventPosting">
  <xsql:query connection="demo" xmlns:xsql="urn:oracle-xsql">
    SELECT name, origin
    FROM location loc
    WHERE loc.origin.x = {@x-coord}
  </xsql:query>
</xsql:event-set>
```

Behind the scenes, the XSQL page processor looks for action elements from the XSQL namespace and for each one found it invokes an appropriate action element handler class to process the element. We have implemented a Java handler for the `<xsql:event-set>` action element named `XSQLEventSetHandler` which is invoked every time such an action element is found in an XSQL page. This handler takes as input the root node of the DOM document fragment resulting from processing the internal XSQL action elements (in our example, the XML document produced by our query), then applies the XSL transformation defined in its “transform” attribute, and finally sends the result (a collection of events represented as XML documents) to an external entity or service in charge of handling subscriptions and notifications of events. Of course, by doing so, we are able to integrate external event services, but the ultimate reason for this last step is that we achieve better scalability by separating both processes: event generation and event notification.

## 2.2 Event Notification Model

So far we have described a collection of data servers that are able to generate event instances in a powerful way, but obviously this is only useful if it serves as an integration mechanism in a many-to-many communication schema. Thus, there must be parties (which can be other data servers or even end-users) that must express their interest in certain types of events (event selection) so that whenever such an event is published a notification is received (notification delivery) by all interested parties. So, the primary challenge of an event notification model is to define the mechanisms by which those two processes take place.

### Event Selection

One of the key aspects that characterize an event selection model is the expressiveness of the language used for subscriptions. As pointed out in [12], there are two factors that influence expressiveness: the visibility that a subscription has on the contents of events (Scope), and the number of operators that can be used in forming subscriptions predicates (Power). There are certainly very powerful formal languages and inference engines that would be capable of expressing highly complex selection expressions over events attributes. However, the more powerful the language is for expressing filters, the less efficient it is the delivery of events in a highly distributed environment [12]. Learning from this lesson, we should look for a language with the following properties: unlimited scope (i.e. the internals of events should be visible in the language), limited power (restricting the number of operators and valid expres-

sions), and homogeneity with the event description language (so that filters can be easily expressed in terms of events expressions). Having these properties in mind we have decided to define filters in terms of XPath [13] expressions that will be evaluated against the events represented as XML documents. XPath expressions are a standard way of addressing parts of an XML document that can also be used for matching. These matching facilities may be used for filtering purposes by defining XPath expressions that match XML events containing a particular node. To illustrate our approach, two examples of filtering expressions that can be obtained with such a language are shown here: a filter matching every `CoordinateReadEvent` event (regardless of its content), and a filter matching `FlightArrivalEvent` events related to flights operated by Iberia arriving either from Valencia or Barcelona:

```
<FILTER expr="/[@Name='CoordinateReadEvent']"/>
<FILTER expr="/[@Name='FlightArrivalEvent']|
/BODY/[AIRLINE='Iberia']|/BODY/[ORIGIN='VLC' or ORIGIN=
'BCN']"/>
```

Basically, events are evaluated against such XPath expressions; if the evaluation process results in a non empty set of sub-nodes, then the event is considered as matching the corresponding filter. A detailed description of possible XPath expressions is out of the scope of this paper, but we claim that the expressiveness of this language is powerful enough to define complex filtering mechanisms.

## Event Subscription

Filters, however, are not the only components needed to describe subscriptions; some parameters characterizing the requester and the delivery are also critical: the identity of the client that will get the notification, the physical handlers to process all notifications for a given client<sup>1</sup>, the number of retries in case of handler unavailability, and the policy to be enforced when delivery finally fails. These delivery requirements are expressed as part of a subscription as an XML document as follows:

```
<SUBSCRIBE>
<HANDLER url=http://.../handler.xsql>
  <RETRY number=10 every=100000 backup=store></RETRY>
</HANDLER>
<HANDLER url=tcp://mymachine:40000>
  <RETRY number=3 every=1000 backup=discard></RETRY>
</HANDLER>
<FILTER>...</FILTER>
</SUBSCRIBE>
```

In the previous example, events that are matched with the specified filter will be notified to a client having two handlers. The first one is an XSQL based handler that will receive the XML description of the event as an input. This information will be used as an input during the processing of the invoked XSQL page (see [10] for more information on processing input XML documents in XSQL pages). In the second case, the XML stream is delivered to a given TCP port making the delivery process flexible to accommodate new protocols. Notice that other protocols could be specified. Every subscription has an associated identifier (a 64 byte UUID identifier like the ones used in JXTA [14] to uniquely identify peers). This UUID can be used by the

---

<sup>1</sup> Note that separating client identities and physical handlers allows for client mobility.

client to unsubscribe notifications for a given handler, or alternatively for all handlers if no handler is specified. This is expressed as well in XML as follows:

```
<UNSUBSCRIBE ID= 1448769379829432795L>
<HANDLER url=http://.../handler.xsql>
</UNSUBSCRIBE>
```

Once the language for describing (un)subscriptions has been presented, we will elaborate on how they are handled within our XSQL based implementation. An XSQL server that is able to process subscriptions must provide a collection of XSQL pages for this purpose. In order to do this, we have created a new XSQL action element called `<xsql-subscription>`. Such a page looks quite simple:

```
<?xml version="1.0"?>
<xsql:subscription type={@type}>
</xsql:subscription>
```

Any client with the intention to un/subscribe just has to invoke such a page and present the XML document describing the un/subscription :

```
http://.../xsql/subscription.xsql?type=unsubscribe
http://.../xsql/subscription.xsql?type=subscribe
```

The handler associated to such action element (XSQLSubscriptionHandler) takes the request, processes the incoming XML un/subscription document and returns an XML document containing the result of the request:

```
<SUBSCRIPTION type="subscribe" status="OK" code=0>
  <Identifier>1448769379829432795L </Identifier>
</SUBSCRIPTION>
<SUBSCRIPTION type="unsubscribe" status="ERROR" code=10>
<Message> The unsubscription identifier
  1448769379829432795L is invalid
</Message>
</SUBSCRIPTION>
```

The mechanism described here requires proper authentication of the requesting clients. In order to comply with this important requirement, we have already defined a security model for XSQL servers based on X509 certificates[15], but the details are outside the scope of this paper.

## Notification Delivery

Notification delivery strategies vary in complexity depending on the type of service to be built. Clearly, a generic distributed notification service, where clients interact locally with a single server but are eligible to receive notifications of events published in other remote servers, is the most difficult problem. As pointed out in [12] the interconnection topology of servers, the routing algorithm and the processing strategy to optimize message traffic are the three critical design issues. A solution to this problem based on covering relations among filters is proposed in SIENA [12], a wide-area notification service that takes into account scalability issues to support a big number of subscribers.

Grid collective services, however, are usually distributed systems with a potentially large number of publishers but a reduced number of event subscribers. In other words, we are interested in connecting asynchronously a reduced number of data servers that hold a big amount of information. Therefore, given that these systems are data-

intensive, the number of events to be communicated may be potentially enormous, but the number of interested parties on those events will usually be much lower. Having this in mind, it was not so critical for this work to build a generic distributed notification service, rather to find a flexible way to interconnect asynchronously data servers and a flexible mechanism to generate domain specific event vocabularies. Nevertheless, as pointed out earlier, filters as defined in this paper satisfy certain covering relations like the one presented in [12]. This means that our model could be used to implement a generic notification service by adding the processing algorithms proposed there to our `XSQLSubscriptionHandler` class. For this early implementation of our model we evaluate and match subscriptions locally within each server and notify events to the handlers associated with the matched filters. Note that we allow handlers to be other XSQL servers that can also act as subscription handlers for other clients. This way we can still achieve reasonable scalability with networks of XSQL subscription handlers.

### 3 Related Works

In the last few years, models, languages and middleware for coordination have been proposed in which XML-based technology or security issues play a central role. Nevertheless, not many proposals deal jointly with both issues as shown next.

LIME[16] offers strong support for coordination in ad-hoc networks, where computational units are mobile entities. This model addresses both logical and physical mobility, i.e. agents or physical devices. Each unit has its own tuple space whose behaviour can be programmed, similar to MARS-X but more restricted because it provides better control over the accesses. The coordination among co-located entities is allowed by automatically merging their tuple spaces. Security enforcement is achieved by implementing authentication, cryptography, and access control in the underlying middleware. However, it disregards any interoperability issues.

TuCSon[17] defines a coordination model based on logic tuple spaces, called *tuple centres*, associated and developed for the coordination of knowledge-oriented mobile agents. These tuple centres are programmable to provide a behaviour specified by the interaction rules. Furthermore, access control policies, such as authentication and authorization, have been implemented on these tuple centres. However, some problems arise regarding interoperability.

Finally, event-driven systems such as SIENA, Gryphon, Elvin and JEDI although providing mechanisms for wide-area notifications do not provide security mechanisms that scale with the potentially vast number of grid users in a V.O. and do not provide flexible mechanisms to accommodate new event vocabularies with negligible programming efforts.

The main advantage exhibited by our approach with respect to the others shown above is its ability to deal with interoperability by using standard protocols and services which makes it really simple to deploy a shared space. Since coordination is XML-based no special adding is needed to the involved nodes. Furthermore, security enforcement is accomplished by standard technologies that allow us not only to define authorization policies but also deal with authentication, privacy and integrity. Besides, the mechanism for defining access policies is scalable with the number of potential processes that may access a shared space.

## 4 Conclusions and Future Work

In this paper we have presented our work on a distributed event model for grid collective services. These systems are usually data intensive distributed repositories that need asynchronous communication. Existing solutions to this problem propose asynchronous communication layers which are domain dependent (the vocabulary for events is previously fixed) and whose implementation is tightly bound to the middleware providing the collective service. These approaches clearly lack flexibility and do not take advantage of a generic event model that is extensible. Our approach, however, proposes a flexible mechanism based on XSL transformations to define event languages, integrates easily data manipulation with event production by means of XSQL pages, requires no programming, provides a security layer, and its implementation uses technologies that are free and in most cases open sourced. All these features make our model suitable for deployment in real production grid projects. Our future work will implement a grid collective service for navigation of distributed RDF graphs which is auto-adaptive and copes with distributed modifications by multiple stakeholders. We also want to explore the applicability of our model to implement distributed workflow management systems for multi-enterprise process management and filtering based added-value services for digital libraries.

## References

1. G. Pfister, *In search of Clusters*, 2nd ed., Prentice Hall, 2001
2. R. Meier, *State of the Art Review of Distributed Event Models*, 2000  
<ftp://ftp.cs.tcd.ie/pub/tech-reports/reports.00/TCD-CS-2000-16.pdf>
3. I. Foster, C. Kesselman, (eds.) *The Grid: Blueprint for a New Computing Infrastructure*. Morgan Kaufmann, 1999
4. I. Foster, C. Kesselman, S. Tuecke, *The Anatomy of the Grid*. Intl. J. Supercomputer Applications, 15(3), 2001
5. K. Czajkowski, S. Fitzgerald, I. Foster, C. Kesselman, *Grid Information Services for Distributed Resource Sharing*. Proceedings Proc. of the Tenth IEEE International Symposium on High-Performance Distributed Computing (HPDC-10), IEEE Press, August 2001
6. W. Hoschek, J. Jaen, A. Samar, H. Stockinger, K. Stockinger, *Data Management in an International Data Grid Project*. Proc. 1st IEEE/ACM International Workshop on Grid Computing, 2000, Springer Verlag
7. M. Thompson, W. Johnston, S. Mudumbai, G. Hoo, K. Jackson, A. Essiari *Certificate-based Access Control for Widely Distributed Resources*. Proc. 8th Usenix Security Symp., 1999
8. A. Chervenak, I. Foster, A. Iamnitchi, C. Kesselman, M. Ripeanu *Giggle: A Framework for Constructing Scalable Replica Location Services*. Working Draft, 2001
9. *Data Management Work Package, D2.2 Architecture and Design Document*, 2001  
[http://grid-data-management.web.cern.ch/grid-data-management/docs/DataGrid-02-D2.2-0103-1\\_2.pdf](http://grid-data-management.web.cern.ch/grid-data-management/docs/DataGrid-02-D2.2-0103-1_2.pdf)
10. S. Muench, *Building Oracle XML Applications*, O'Reilly, 2000
11. W. Smith, D. Gunter, D. Quesnel *Simple XML Producer/Consumer Protocol*. Global Grid Forum Grid Working Draft, 2001  
<http://www.didc.lbl.gov/GGF-PERF/GMA-WG/papers/GWD-GP-8-2.pdf>

12. A. Carzaniga, D.S. Rosenblum, and A.L. Wolf. Achieving Expressiveness and Scalability in an Internet-Scale Event Notification Service. Nineteenth ACM Symposium on Principles of Distributed Computing (PODC2000), Portland OR(July, 2000)
13. The XML Path Language <http://www.w3.org/TR/xpath>
14. Sun Microsystems, JXTA v1.0 Protocols Specification  
<http://spec.jxta.org/v1.0/docbook/JXTAProtocols.html>
15. J. Jaen., J.H.Canós. An Advanced Security Infrastructure for Heterogeneous Relational Grid Data Services. 1st European Across Grids Conf., 2003, LNCS 2970 Springer 2004
16. G.-C. Roman, R. Handorean. Secure Sharing of Tuple Spaces in Ad Hoc Settings. Proc. Int. Workshop on Security Issues in Coordination Models, Languages, and Systems, ENCS Vol. 85, No. 3. (2003)
17. M. Cremonini, A. Omicini, F. Zambonelli.: Multi-Agent Systems on the Internet: Extending the Scope of Coordination towards Security and Topology. Workshop on Modelling Autonomous Agents in a Multi-Agent Worlds (1999), Valencia (E), LNAI n. 1647, (1999)

# Symbolic Agent Negotiation for Semantic Web Service Exploitation

Peep Küngas<sup>1</sup>, Jinghai Rao<sup>1</sup>, and Mihhail Matskin<sup>2</sup>

<sup>1</sup> Norwegian University of Science and Technology  
Department of Computer and Information Science  
Trondheim, Norway

{peep,jinghai}@idi.ntnu.no

<sup>2</sup> Royal Institute of Technology  
Department of Microelectronics and Information Technology  
Kista, Sweden  
misha@imit.kth.se

**Abstract.** This paper presents an architecture and a methodology for agent-based Web service discovery and composition. We assume that Web services are described with declarative specifications like DAML-S. Based on the declarative information about services, symbolic reasoning can be applied while searching for or composing automatically new services.

We propose that symbolic agent negotiation could be used for dynamic Web service discovery and composition. Symbolic negotiation, as we consider it here, is a mixture of distributed planning and information exchange. Therefore, by using symbolic negotiation for automated service composition, we support information collection and integration during service composition. The latter aspect has been largely neglected in automated service composition until now.

## 1 Introduction

Several initiatives in Web services provide platforms and languages that should allow easy integration of heterogeneous systems. In particular, such standards as UDDI, WSDL, SOAP and a part of DAML-S ontology define standard ways for service discovery, description and invocation. Some other initiatives such as BPEL4WS and DAML-S ServiceModel, are focused on representing service composition where flow of a process and bindings between services are known a priori.

The ability to efficiently select and integrate inter-organisational and heterogeneous Web services at runtime is an important requirement to the Web service provision. In particular, if no single Web service can satisfy the functionality required by a user, there should be a possibility to combine existing services together in order to fulfill the request. A challenge is that Web services can be created and updated on the fly and it may be beyond human capabilities to analyze the required services and compose them manually. As a result,

the service composition becomes a complex problem, which definitely cannot be solved without some degree of automation.

Several articles address automatic composition of Web services [10, 13, 14, 16]. However, they all require existence of a central directory of Web service specifications, which contrast largely to the dynamic nature of the Web. In the Web the set of available services changes rapidly – new services are created, old ones are modified or removed. Keeping track of all these changes is a huge burden for a centralised directory. Some essential issues in decentralised Web service provision have been addressed by Papazoglou et al [11].

Another disadvantage of a centralised approach is that it only allows service requesters to locate services, while service providers lack an ability to attract potential customers. The agent-based architecture we propose here gives service providers a more proactive role in the service composition process. Our service provision architecture is based on the multi-agent system AGORA [8], which provides an infrastructure, where service providers and requesters can meet with each-other.

We propose that symbolic agent negotiation could be used as a mechanism for discovering available Web services and composing new ones automatically. If no service, satisfying user's requirements, is found, symbolic negotiation between agents is initiated and a new composite Web service is constructed dynamically. We take advantage of symbolic negotiation framework described by Küngas and Matskin [4, 5], where linear logic [3] (LL) is applied to agent negotiation.

The work presented in this article is closely related to framework in [12], where LL theorem proving is applied for automated Web service synthesis in a centralised manner. In this article we go beyond the centralised approach and propose how to take advantage of intelligent agent technologies for distributed Web service composition. We assume that service descriptions have been already translated into LL formalism. Therefore we do not describe mapping from DAML-S to LL formulae.

The rest of the paper is organised as follows. In Section 2 we give a general description of the system architecture. Section 3 formalises symbolic negotiation for agent systems. Section 4 demonstrates usage of symbolic negotiation for Web service composition. Section 5 reviews the related work and Section 6 concludes the paper.

## 2 The System Architecture

The AGORA multi-agent environment [8] was developed with the intention to support cooperative work between agents. The system consists of 2 types of components (nodes) – agents and agoras. Agoras are cooperative nodes which facilitate agent communication, coordination and negotiation. Moreover, agoras encapsulate a method for (partial) service composition. An agora node contains *default agents* and *registered agents*. In our scenario, the default agents are Agora Manager and Negotiator. Agora Manager implements general agora functions,

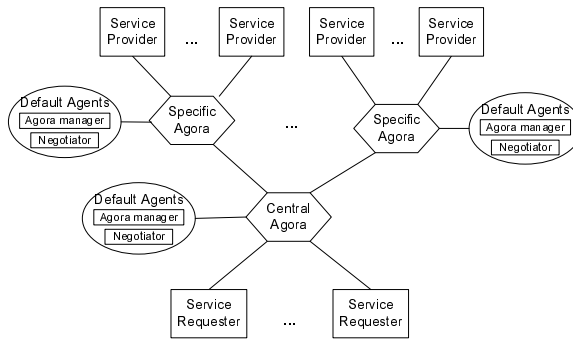


such as service matchmaking and agent/service registration, while Negotiator applies symbolic negotiation.

Service matchmaking basically involves finding an atomic service, which satisfies agent requirements for a service. Negotiator applies symbolic negotiation for composing new services automatically. Negotiation is applied, if Agora Manager failed to find an atomic service satisfying agents' requirements.

Service provider agents register their services at specific agoras according to their service domains. For example agents providing services for selling, buying and managing hardware register themselves and available services at agoras devoted to hardware. Specific agoras may represent also coalitions of service providers.

Service requester agents, however, register requests for services at the central agora. Then the central agora negotiates with specific agoras to find services satisfying requester agents' requirements. The central agora also mediates information exchange between different requester agents. Moreover, the central agora might also form requester agent coalitions, if it is needed for certain services. It may happen for instance that a (composite) service requires input from more than one service requester agent. Our system architecture is depicted in Fig. 1.

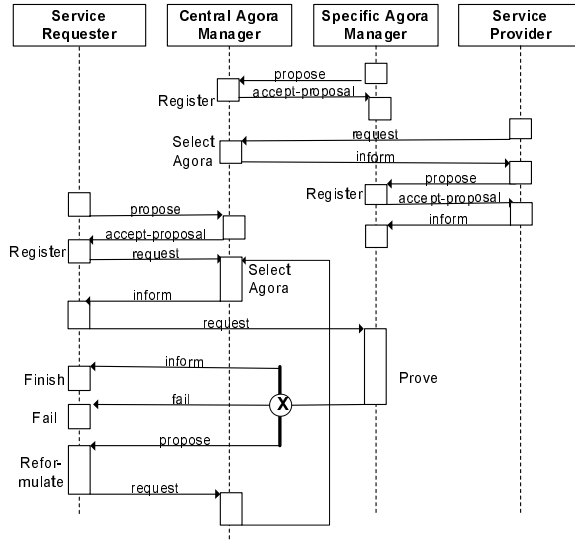


**Fig. 1.** The system architecture.

A specific service composition architecture can be specified through refinement and instantiation of the generic architecture described above. An instance of the generic architecture is elaborated in Section 4 and depicted in Fig. 3.

Fig. 2 presents the proposed interaction protocol in Agent UML notation. This protocol uses the FIPA (The Foundation for Intelligent Physical Agents) reserved communicative acts, allowing thus interoperability with other FIPA compliant agent systems. The agent interaction process is summarised in the following.

The central agora is a search control point for requester agents' queries. Specific agoras are cooperative nodes gathering agents who provide the same sort of services. Specific agoras register themselves to the central agora. After receiving the registration request from a service provider agent, the central agora



**Fig. 2.** The interaction protocol.

locates a specific agora where the service provider could register itself. In our case, the service providers registered to the same specific agora provide services in the same domain.

Service requester agents register themselves to the central agora. Additionally they publish their requirements in a declarative language. After that the central agora tries to locate an atomic service satisfying the requirements. However, if no suitable atomic service is found, symbolic negotiation is initiated. During symbolic negotiation the central agora contacts specific agoras to receive a composite service for satisfying the particular requirements.

It might happen that services from different agoras are needed for a composition. In that case the central agora receives a partial composition from one specific agora and forwards it to another specific agora for further composition. The process is called symbolic negotiation – the central agora negotiates with specific agoras to compose a solution for a service requester agent. Finally a composite service is constructed and returned to the requester agent, who initiated the symbolic negotiation.

### 3 Symbolic Negotiation

In this section we present formal foundations of symbolic agent negotiation as it was proposed in [4, 5]. Symbolic negotiation is generally defined as an interactive process involving partial deduction (PD) and agent communication. Throughout this paper we consider PD only for propositional LL, since Web services in DAML-S can be represented in a propositional LL fragment. However, the extension in [5] allows usage of first-order LL as well for symbolic negotiation.

For representing symbolic negotiation we consider !Horn fragment of LL (HLL) consisting of multiplicative conjunction ( $\otimes$ ), linear implication ( $\multimap$ ) and “of course” operator (!). From symbolic negotiation point of view the logical expression  $A \vdash C$  means that an agent can provide  $A$  and requires  $C$ . ! represents unbounded access to resources. Thus ! $A$  means that resource  $A$  can be used as many times as needed. To increase the expressiveness of formulae, we are using in the following abbreviation  $a^n = \underbrace{a \otimes \dots \otimes a}_n$ , for  $n \geq 0$ , with the degenerate case  $a^0 = 1$ .

An agent is presented with the following LL sequent  $\Gamma; S \vdash G$ , where  $\Gamma$  is a set of extralogical LL axioms representing agent’s capabilities (services),  $S$  is the initial state and  $G$  is the goal state of an agent. Both  $S$  and  $G$  are multiplicative conjunctions of literals. Every element of  $\Gamma$  has the form  $\vdash I \multimap O$ , where  $I$  and  $O$  are formulae in conjunctive normal form which are, respectively, consumed and generated when a particular capability is applied. It has to be mentioned that a capability can be applied only, if conjuncts in  $I$  form a subset of conjuncts in  $S$ .

Messages are defined with the tuple  $(id_{req}, S, R, O)$ , where  $id_{req}$ ,  $S$ ,  $R$  and  $O$  denote respectively message identifier, sender, receiver and offer. Message identifier is needed to keep track of different negotiations. Sender and receiver are identifiers of participating agents and offer is represented with a LL sequent in form  $A \vdash B$ , where  $A$  and  $B$  are multiplicative conjunctions of literals.

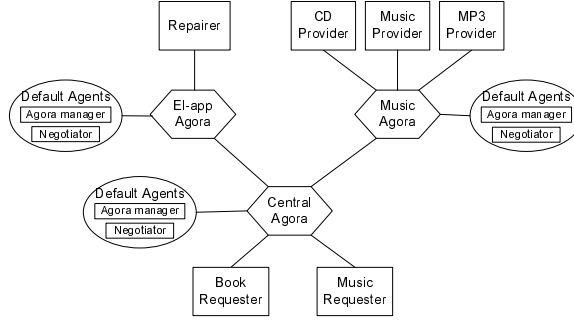
In [4] PD steps  $\mathcal{R}_b(L_i)$  and  $\mathcal{R}_f(L_i)$  were defined for back- and forward chaining:

$$\frac{S \vdash B \otimes C}{S \vdash A \otimes C} \mathcal{R}_b(L_i) \quad \frac{A \otimes C \vdash G}{B \otimes C \vdash G} \mathcal{R}_f(L_i)$$

$L_i$  in the inference figures is a labelling of a particular LL axiom representing an agent’s capability (computability clause in PD) in the form  $\vdash B \multimap_{L_i} A$ .  $\mathcal{R}_f(L_i)$  and  $\mathcal{R}_b(L_i)$  apply clause  $L_i$  to move the initial state towards the goal state or the other way around.  $A$ ,  $B$  and  $C$  are formulae in HLL.

## 4 An Example

In this section we demonstrate the usage of symbolic negotiation in distributed service composition. Before we start the example, we have to emphasize two assumptions. First, we assume the users use standard Web service languages, e.g. DAML-S to specify the Web services. The logical formulas used in the example can be translated from DAML-S by the method introduced in [12]. Second, the Web services here include both information-gathering services and world-altering services. DAML-S has capability to distinguish the two kinds of services [10]. In general, the “input” and “output” properties provide information about dataflow, while the “precondition” and “effect” properties tell what the Web service actually does. Hence the Web service in the example not only can present the information flow, but also can indicate the exchange of the goods, such as CD player and books.



**Fig. 3.** The example architecture.

The particular system architecture and its components are depicted in Fig. 3. In our scenario we have two requester agents –  $\mathcal{R}_1$  and  $\mathcal{R}_2$ . The goal of  $\mathcal{R}_1$  is to listen a music (*Music*). Initially  $\mathcal{R}_1$  has a book (*Book*), a broken CD player (*BrokenCDPlayer*) and 5 dollars (*Dollar*<sup>5</sup>). Goals, resources and capabilities of  $\mathcal{R}_1$  are described in LL with the following formulae.

$$G_{\mathcal{R}_1} = \{Music\}, \quad S_{\mathcal{R}_1} = \{Book \otimes BrokenCDPlayer \otimes Dollar^5\}, \quad \Gamma_{\mathcal{R}_1} = \emptyset.$$

Another query agent  $\mathcal{R}_2$  is looking for a book (*Book*) and is in possession of 10 dollars (*Dollar*<sup>10</sup>). Goals, resources and capabilities of the query agent  $\mathcal{R}_2$  are described in LL with the following formulae.

$$G_{\mathcal{R}_2} = \{Book\}, \quad S_{\mathcal{R}_2} = \{Dollar^{10}\}, \quad \Gamma_{\mathcal{R}_2} = \emptyset.$$

In addition we have several service provider agents. However, since they published their services through particular agoras and these agoras take care of advertising the services, we do not list here the internal states of the service provider agents. Instead we present the internal states of agoras. Although agoras may have their own resources and goals, which determine their policies, we consider here only a simple case, where agoras take advantage of registered services/capabilities only.

According to the literals service providers can “produce”, the providers are aggregated into two agoras – one for music and another for electrical appliances. In the Music Agora  $\mathcal{M}$ , three agents, *MusicProvider*, *CDProvider* and *MP3Provider*, can provide services related to music. Services *playCD* and *playMP3* provide respectively knowledge about requirements for playing CD-s and MP3-s. Services *buyMP3* and *buyCD* provide means for ordering particular music media.

$$\begin{aligned} \Gamma_{\mathcal{M}} = & \vdash_{MusicProvider} CD \otimes CDPlayer \multimap_{playCD} Music, \\ & \vdash_{MusicProvider} MP3 \otimes MP3Player \multimap_{playMP3} Music, \\ & \vdash_{CDProvider} Dollar^5 \multimap_{buyCD} CD, \\ & \vdash_{MP3Provider} Dollar^3 \multimap_{buyMP3} MP3. \end{aligned}$$

The agora  $\mathcal{E}$  is an aggregation of the agents who can provide electrical appliances. It only advertises one service *repair* from agent *Repairer*. The service description declares that the agent can repair a CD player by charging 10 dollars.

$$\Gamma_{\mathcal{E}} = \vdash_{\text{Repairer}} \text{Dollar}^{10} \otimes \text{BrokenCDPlayer} \multimap_{\text{repair}} \text{CDPlayer}.$$

Let us look now how symbolic negotiation is applied for constructing dynamically services, which satisfy users' goals. Initially the query agent  $\mathcal{R}_1$  sends out a query to agora  $\mathcal{M}$  for finding a service satisfying its requirements:

$$(o_1, \mathcal{R}_1, \mathcal{M}, \text{Book} \otimes \text{BrokenCDPlayer} \otimes \text{Dollar}^5 \vdash \text{Music}).$$

The query would be satisfied by a service

$$\vdash \text{Book} \otimes \text{BrokenCDPlayer} \otimes \text{Dollar}^5 \multimap \text{Music}.$$

Unfortunately the service requirement is too specific and no matching service is found. However, agora  $\mathcal{M}$  modifies the received offer and sends back the following offers:

$$(o_2, \mathcal{M}, \mathcal{R}_1, \text{Book} \otimes \text{BrokenCDPlayer} \vdash \text{CDPlayer})$$

and

$$(o_3, \mathcal{M}, \mathcal{R}_1, \text{Book} \otimes \text{BrokenCDPlayer} \otimes \text{Dollar}^2 \vdash \text{MP3Player}),$$

which were deduced through PD in the following way:

$$\frac{\frac{\frac{\text{Book} \otimes \text{BrokenCDPlayer} \vdash \text{CDPlayer}}{\text{Book} \otimes \text{BrokenCDPlayer} \otimes \text{Dollar}^5 \vdash \text{Dollar}^5 \otimes \text{CDPlayer}} \text{normalise}}{\text{Book} \otimes \text{BrokenCDPlayer} \otimes \text{Dollar}^5 \vdash \text{CD} \otimes \text{CDPlayer}} \mathcal{R}_b(\text{buyCD})}{\text{Book} \otimes \text{BrokenCDPlayer} \otimes \text{Dollar}^5 \vdash \text{Music}} \mathcal{R}_b(\text{playCD})$$

$$\frac{\frac{\frac{\text{Book} \otimes \text{BrokenCDPlayer} \otimes \text{Dollar}^2 \vdash \text{MP3Player}}{\text{Book} \otimes \text{BrokenCDPlayer} \otimes \text{Dollar}^5 \vdash \text{Dollar}^3 \otimes \text{MP3Player}} \text{normalise}}{\text{Book} \otimes \text{BrokenCDPlayer} \otimes \text{Dollar}^5 \vdash \text{MP3} \otimes \text{MP3Player}} \mathcal{R}_b(\text{buyMP3})}{\text{Book} \otimes \text{BrokenCDPlayer} \otimes \text{Dollar}^5 \vdash \text{Music}} \mathcal{R}_b(\text{playMP3})$$

where *normalise* is another LL inference figure, which reduces the number of literals from a particular sequent:

$$\frac{\frac{\overline{A \vdash A} \quad Id \quad \overline{B \vdash C}}{B, A \vdash C \otimes A} R\otimes}{B \otimes A \vdash C \otimes A} L\otimes$$

Agent  $\mathcal{R}_1$  chooses the offer  $o_2$  and forwards it to the electrical appliance agora  $\mathcal{E}$ :

$$(o_4, \mathcal{R}_1, \mathcal{E}, \text{Book} \otimes \text{BrokenCDPlayer} \vdash \text{CDPlayer}).$$

Agora  $\mathcal{E}$  modifies the offer further and ends up with the following counteroffer:

$$(o_5, \mathcal{E}, \mathcal{R}_1, \text{Book} \vdash \text{Dollar}^{10}),$$

which was derived in the following way:

$$\frac{\frac{Book \vdash Dollar^{10}}{Book \otimes BrokenCDPlayer \vdash Dollar^{10} \otimes BrokenCDPlayer} \text{ normalise}}{Book \otimes BrokenCDPlayer \vdash CDPlayer} \mathcal{R}_b(\text{repair})$$

Since no service provider can produce the *Dollar* literal, the message is sent to the central Agora, which has an overview of requester agents' requirements. Fortunately, it turns out that agents  $\mathcal{R}_1$  and  $\mathcal{R}_2$  can satisfy mutually their requirements such that requester  $\mathcal{R}_1$  gets 10 dollars and requester  $\mathcal{R}_2$  gets a book. The resulting service composition can be translated to a process description languages like DAML-S process model or BPEL4WS. The exact translation process is described in another paper and is not covered here.

## 5 Related Work

Gibbins et al [2] demonstrated, through an implementation, the usage of DAML-S Web service descriptions within agents. Their agents embody DAML-S descriptions of Web services and agent communication is managed through FIPA ACL.

Another step towards incorporating Web services into agents is proposed by Ardissono et al [1]. Their main contribution is support for more rigorous Web service execution protocols compared to currently prevalent 2-step protocols (sending input data and then collecting results). We go beyond that approach by allowing a server agent to compose a sequence of action for a service consumer such that a required sequence of service command executions is constructed automatically at server side and the consumer can provide all details once.

In [10] a modification of Golog [6] programming language is used for automatic construction of Web services. It is argued there that Golog provides a natural formalism for automatically composing services on the Semantic Web. Golog has been enriched with some extralogical constructions like **if**, **while**, etc.

Waldinger [15] proposes initial ideas for another deductive approach. The approach is based on automated deduction and program synthesis and has its roots to work presented in [7]. First available services and user requirements are described with a first-order language, related to classical logic, and then constructive proofs are generated with SNARK theorem prover.

In [16] SHOP2 planner is applied for automatic composition of DAML-S services. Other planners for automatic Web service construction include [13, 9]. Thakkar et al. [14] consider dynamic composition of Web services using mediator agent architecture. The mediator takes care of user queries, generates wrappers around information services and constructs a service integration plan.

## 6 Conclusions and Further Work

In this paper we described an agent architecture for automatic composition of Web services. Agent-specific aspects provide Web service composition with

proactivity, reactivity, social ability and autonomy, while usage of DAML-S, FIPA ACL and application domain specific ontologies provide a standardised medium for Web service deployment. Usage of DAML-S allows publishing semantically enriched specifications of Web services and thus fits well to the Semantic Web initiative, where both Web services and data are labelled with semantic information.

Given that services are represented in DAML-S, they are translated to LL formulae for internal agent reasoning. Given such representation, agents can employ symbolic agent negotiation for composing new Web services according to their requirements. This approach leads to distributed composition of Web services.

We have implemented a symbolic negotiation framework, which exploits an AI planner called RAPS for symbolic reasoning and automated Web service composition. Additionally we have developed a tool for automatically composing Web services whose descriptions are given in DAML-S. The system applies RAPS planner for centralised Web service composition. Further work would concentrate to the evaluation of decentralised Web service composition. Additionally, we would like to extend our method for service composition to exploit also business models, which specify additional relationships between services.

## Acknowledgments

This work was partially supported by the Norwegian Research Foundation in the framework of the Information and Communication Technology (IKT-2010) program – the ADIS project. The authors would also thank the anonymous reviewers for their constructive comments.

## References

1. L. Ardissono, A. Goy, and G. Petrone. Enabling conversations with web services. In *Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS 2003, July 14–18, 2003, Melbourne, Victoria, Australia*, pages 819–826. ACM Press, 2003.
2. N. Gibbins, S. Harris, and N. Shadbolt. Agent-based semantic web services. In *Proceedings of the Twelfth International World Wide Web Conference, WWW2003, Budapest, Hungary, May 20–24, 2003*, pages 710–717. ACM Press, 2003.
3. J.-Y. Girard. Linear logic. *Theoretical Computer Science*, 50:1–102, 1987.
4. P. Küngas and M. Matskin. Linear logic, partial deduction and cooperative problem solving. In *Proceedings of the First International Workshop on Declarative Agent Languages and Technologies (in conjunction with AAMAS 2003), DALT'2003, Melbourne, Australia, July 15, 2003*. Springer-Verlag, 2004. To appear.
5. P. Küngas and M. Matskin. Symbolic negotiation with linear logic. In *Proceedings of the Fourth International Workshop on Computational Logic in Multi-Agent Systems, CLIMA IV, Fort Lauderdale, FL, USA, January 6-7, 2004*. Springer-Verlag, 2004. To appear.

6. H. J. Levesque, R. Reiter, Y. Lespérance, F. Lin, and R. B. Scherl. Golog: A logic programming language for dynamic domains. *Journal of Logic Programming*, 31(1–3):59–83, 1997.
7. Z. Manna and R. J. Waldinger. A deductive approach to program synthesis. *ACM Transactions on Programming Languages and Systems*, 2(1):90–121, 1980.
8. M. Matskin, O. J. Kikkeluten, S. B. Krossnes, and Østein Sæle. Agora: An infrastructure for cooperative work support in multi-agent systems. In T. Wagner and O. F. Rana, editors, *International Workshop on Infrastructure for Multi-Agent Systems, Barcelona, Spain, June 3–7, 2000, Revised Papers*, volume 1887 of *Lecture Notes in Computer Science*, pages 28–40. Springer-Verlag, 2001.
9. D. McDermott. Estimated-regression planning for interaction with web services. In *Proceedings of the 6th International Conference on AI Planning and Scheduling, Toulouse, France, April 23–27, 2002*. AAAI Press, 2002.
10. S. McIlraith and T. C. Son. Adapting Golog for composition of semantic web services. In *Proceedings of the Eighth International Conference on Knowledge Representation and Reasoning (KR2002), Toulouse, France, April 22–25, 2002*, pages 482–493. Morgan Kaufmann, 2002.
11. M. P. Papazoglou, B. J. Krämer, and J. Yang. Leveraging web-services and peer-to-peer networks. In J. Eder and M. Missikoff, editors, *15th International Conference on Advanced Information Systems Engineering, CAiSE 2003, June 16–18, 2003, Klagenfurt, Austria*, volume 2681 of *Lecture Notes in Computer Science*, pages 485–501. Springer-Verlag, 2003.
12. J. Rao, P. Küngas, and M. Matskin. Application of linear logic to web service composition. In *Proceedings of the First International Conference on Web Services (ICWS 2003), Las Vegas, USA, June 23–26, 2003*, pages 3–9. CSREA Press, 2003.
13. M. Sheshagiri, M. desJardins, and T. Finin. A planner for composing services described in DAML-S. In *Proceedings of the AAMAS Workshop on Web Services and Agent-based Engineering*, 2003.
14. S. Thakkar, C. A. Knoblock, J. L. Ambite, and C. Shahabi. Dynamically composing web services from on-line sources. In *Proceeding of 2002 AAAI Workshop on Intelligent Service Integration, Edmonton, Alberta, Canada, 2002*.
15. R. Waldinger. Web agents cooperating deductively. In *Proceedings of FAABS 2000, Greenbelt, MD, USA, April 5–7, 2000*, volume 1871 of *Lecture Notes in Computer Science*, pages 250–262. Springer-Verlag, 2001.
16. D. Wu, B. Parsia, E. Sirin, J. Hendler, and D. Nau. Automating DAML-S web services composition using SHOP2. In *Proceedings of the 2nd International Semantic Web Conference, ISWC 2003, Sanibel Island, Florida, USA, October 20–23, 2003*, 2003.



# Object-Oriented Realization of Workflow Views for Web Services – An Object Deputy Model Based Approach\*

Zhe Shan<sup>1</sup>, Zhiyi Long<sup>2</sup>, Yi Luo<sup>3</sup>, and Zhiyong Peng<sup>3</sup>

<sup>1</sup> Department of Computer Engineering and Information Technology,  
City University of Hong Kong, Kowloon, Hong Kong  
zenga@it.cityu.edu.hk

<sup>2</sup> College of information engineering, China Institute of Metrology, Hangzhou, China  
longzhiyi01@hotmail.com

<sup>3</sup> State Key Lab of Software Engineering, Wuhan University, Wuhan, China  
luo\_guo02@hotmail.com, zypeng@public.wh.hb.cn

**Abstract.** Adapted from the concept of views in databases, workflow views are derived from workflows as a fundamental support for workflow inter-operability and visibility by external parties in a web services environment. However, because of the implementation difficulties of views in OODB systems, it is tough to realize workflow views in object-oriented workflow management systems (OO WFMS). In this paper, we adopt the object deputy model to support the realization of workflow views in OO WFMS. The detailed definitions are given based on a reference model of an OO WFMS, namely ADOME-WFMS. Furthermore, we introduce E-ADOME, which enables the implementation of workflow views in the web services environment.

## 1 Introduction

Workflow views [8] are derived from workflows as the fundamental support for workflow inter-operability and visibility by external parties [9][7]. The components of a workflow view include the process flow graph, input/output parameters, objects, rules, events, exceptions and exception handlers, which are also contained in a workflow. Hence, workflow view encloses the information about business process structure and contents. Based on specific business collaboration, a company may derive a corresponding workflow view based on the local workflow system. Such a workflow view includes all the necessary information of the company for this business. It can be used as the interaction interface in a business transaction which is carried out with external parties.

Meanwhile, the need for enhanced flexibility of workflow modeling and execution, and the integration of applications in heterogeneous environments emerged in the workflow context, make the object-oriented workflow management systems (OO WFMSs)

---

\* This research is supported by the State Key Lab of Software Engineering (Wuhan University, China) under grant: SKLSE03-01, NSFC (National Natural Science Foundation of China) (60273072), National 863 Project (2002AA423450), the Research Fund for the Doctoral Program of Higher Education (20010486029) and Bubei Natural Science Foundation for the Distinguished Youth (2002AC003).

one of the promising topics in this area [17][11][18][16]. The OO technology offers WFMSs the power of reusability, flexibility, scalability and persistency. However, objects views, roles and migration are always the difficult issues in an OO database, which makes workflow views hard to realize in OO WFMSs.

In this paper, we adopt the object deputy model [10] to implement workflow views in an OO WFMS. Based on ADOME-WFMS [6], a reference model of OO WFMSs, we specify the components of workflow views, in addition to defining three kinds of workflow views. The resultant OO WFMS is further extended to E-ADOME, which supports the implementation of workflow views in the web services environment.

The remainder of the paper is organized as follows. Section 2 introduces the background of the workflow view and the object deputy model. Section 3 discusses the workflow object model used in this work. Section 4 defines the workflow views via the object deputy model. Section 5 introduces the E-ADOME and the functions of its each layer. Section 6 compares the works related to this paper and Section 7 concludes the paper with plans for further research.

## 2 Background

### 2.1 Workflow View

Motivated by views of databases, [8] proposed the use of workflow views as a fundamental mechanism for cross-organizational workflow interaction. A workflow includes a flow graph of activities and a set of input/output messages. A workflow view is a structurally correct subset of a workflow. The use of workflow views facilitates sophisticated interactions among workflow management systems (WFMSs) and allows these interactions to inter-operate in a gray box mode (i.e., they can access each other's internal information to some extent). The artifact of workflow views is therefore a handy mechanism to enact and enforce cross-organizational inter-operability over the Internet. In addition, workflow views are useful in providing access to business processes for external customers or users, including B2C e-commerce. Even within an organization, workflow views are useful for security applications, such as to restrict the access to a workflow system (resembling the use of views in databases). A meta-model of workflow views in UML class diagram is shown in Figure 1, illustrating their components and relations.

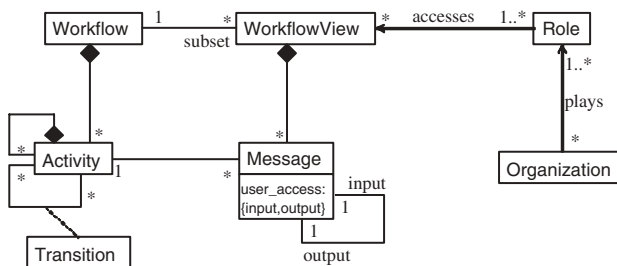
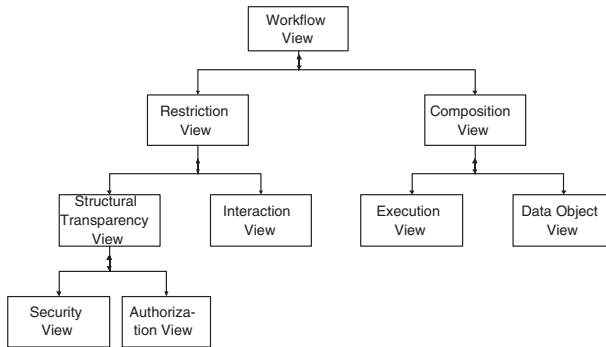


Fig. 1. Workflow View Meta-Model.

Motivated by various situations relating to an e-commerce environment, some representative types of workflow views are illustrated in Figure 2. A *workflow restriction view* is a structurally correct subset of a workflow definition. As shown in Figure 2, there are two specialized view types underneath: *Structural Transparency View* and *Interaction View*. A *Structural Transparency View* is for determining whether the internal details of a workflow are visible by a user, i.e., whether the workflow appears to be a black box (totally invisible), a white box (totally visible) or a grey box (partially visible). Often, *security views* and *authorization views* are based on structural transparency views. An *Interaction View* is the appearance of the workflow to an external user or agent to interact with the workflow. It is the subset of processes that has interaction with the user or agent. A *workflow composition view* is a virtual workflow composed of components from different workflows, which may span across organizational boundaries.



**Fig. 2.** A Taxonomy of Workflow Views.

## 2.2 Object Deputy Model

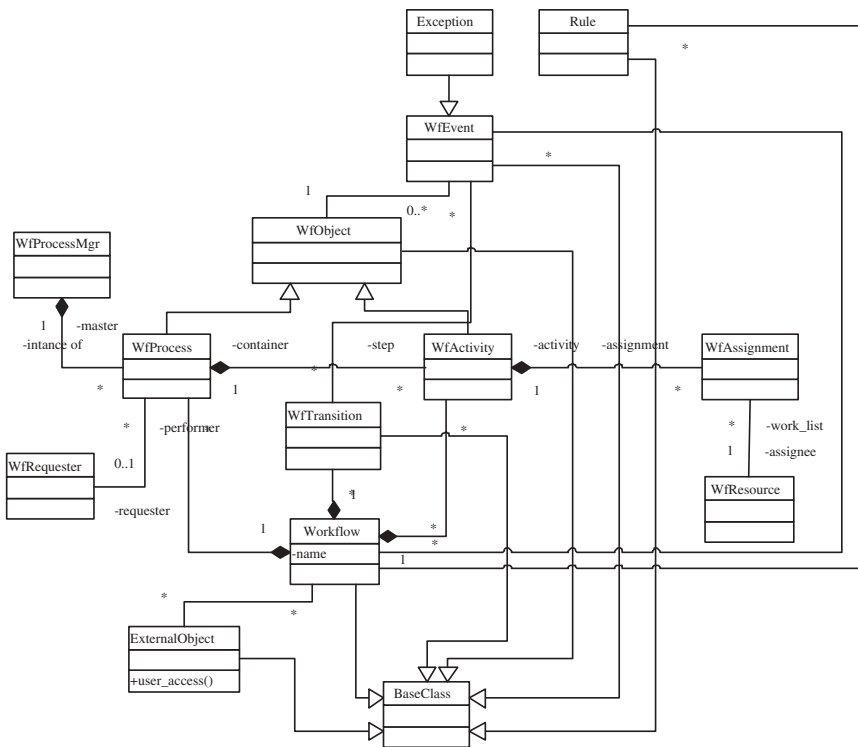
The object deputy model [10] via definition of deputy objects and deputy classes successfully overcomes the disadvantages of traditional object-oriented data models. A deputy object has its own persistent identifier, and may have additional attributes and methods that are not derived from its source objects(s). The attributes and methods of an object can be inherited by its deputy objects. The inheritance is realized through switching operations that can change the names and types of the inherited attributes and methods. There is a bilateral link between an object and one of its deputy objects, which allows not only inheritance but also update propagation between them.

The object deputy model is more flexible than the conventional object-oriented model. It can provide inheritance for specialization, generalization and aggregation. And based on it, the view mechanism is easy to be implemented. In the object deputy model, the object deputy algebra that contains six kinds of algebraic operations is used to manipulate classes or deputy classes. The *Select* operation is used to derive a deputy class of which instances are the deputy objects of the instances of the source class selected according to some selection predicate. The *Project* operation is used to derive a deputy class which allows only part of attributes and methods of the source class to be

inherited. The *Extend* operation is used to derive a deputy class of which instances are extended with additional attributes and methods that can not be derived from the source class. The *Union* operation is used to derive a deputy class of which extension consists of deputy objects of instances of more than one source class. The *Join* Operation is used to derive a deputy class of which instances are deputy objects for aggregating instances of source classes according to some combination predicate. The *Grouping* operation is used to derive a deputy class of which instances are deputy objects for grouping instances of source class according to some grouping predicate.

### 3 Workflow Object Model

Figure 3 depicts the workflow object model used in this work, which is based on the workflow management facility specification proposed by the Object Management Group (OMG)[15].



**Fig. 3.** Workflow Object Model.

*WfRequester* is the interface that has a direct concern with the execution and results of a workflow process - it represents the request for some work to be done. *WfObject* is an abstract base interface that defines common attributes, states, and operations for *WfProcess* and *WfActivity*. Each significant status change of a *WfObject* triggers

a *WfEvent* to be published; *WfEvents* are distributed using the event service, such as CORBA Event Service. A *WfException* is a *WfEvent* that deviates from normal behavior or may prevent forward progress of a workflow. A *WfProcessMgr* represents a template for a specific workflow process; it is used to create instances of a workflow process. A *WfProcess* is the performer of a workflow request, and the *WfProcess* interface specializes a *WfObject* interface by adding an operation to start the execution of the process, an operation to obtain the result produced by the process and relationships with *WfRequester* and *WfActivity*. *WfActivity* is a step in a *WfProcess* and may also be a *WfRequester*. Each *WfActivity* is linked to a *WfResource* which is a person or thing that can do and accept it by a *WfAssignment*; the latter represents the assignment of a particular workflow Participant to a process step. *ExternalObject* defines the objects as the external objects associated with a workflow instance. *WfTransition* is a point during the execution of a process instance where one activity completes and the thread of control passes to another, which starts. Workflow from which workflow views are directly derived from generalizes the entire workflow information.

## 4 Object Deputy Model for Workflow Views

A workflow view actually is defined as views of the objects of the original workflow. Since the object deputy model can easily realize the object view, a workflow view can be defined via the definition of the four kinds of deputy classes, the latter are derived from the classes in the original workflow as the source classes.

A formal definition of objects and classes is given as follows.

**Definition 1.** A class named as *C* is represented as

$$C = \langle \{o\}, \{T_a : a\}, \{m : \{T_p : p\}\} \rangle.$$

1.  $\{o\}$  is the extent of *C*, where *o* is one of instances of *C*.
2.  $\{T_a : a\}$  is the set of attribute definitions of *C*, where *a* and *T<sub>a</sub>* represent the name and type of an attribute, respectively. The value of attribute *a* of object *o* is expressed by *o.a*. For each attribute *T<sub>a</sub> : a*, there are two basic method: *read(o, a)* for reading *o.a* and *write(o, a, v)* for writing *o.a* with the new value *v*.
3.  $\{m : \{T_p : p\}\}$  is the set of method definitions of *C*, where *m* and  $\{T_p : p\}$  denote a method name and a set of parameters. *p* and *T<sub>p</sub>* represent a parameter name and type, respectively.

### 4.1 Components of Workflow Views

The components of a workflow include the process flow graph, input/output parameters, objects, rules, events, exceptions and exception handlers associated with the workflow. A workflow view is a structurally correct subset of a workflow definition. Thus, a view for a workflow also contains these components, as stipulated below.

**Process Flow Graph.** Let the original process be the source class. A process in the flow graph is the deputy class that derived from the original process as the source class by the Project and Extend operation, which may inherit all, part or none of the attributes and methods of the source class; this implements the white-box, black-box, or grey box, respectively. Some additional attributes and methods may be extended. The formal definition is presented as follows.

**Definition 2.** Let  $C^{s_1} = \langle \{o^{s_1}\}, \{T_{a^{s_1}} : a^{s_1}\}, \{m^{s_1} : \{T_{p^{s_1}} : p^{s_1}\}\} \rangle$  be a source class that represents the original process,  $\{T_{a_-^{s_1}} : a_-^{s_1}\}$  and  $\{m_-^{s_1} : \{T_{p_-^{s_1}} : p_-^{s_1}\}\}$  be subsets of attributes and methods of  $C^{s_1}$  which are allowed to be inherited by the class  $C^{d_1}$  that represents the process in the flow graph of the workflow view,  $\{T_{a_+^{d_1}} : a_+^{d_1}\}$  and  $\{m_+^{d_1} : \{T_{p_+^{d_1}} : p_+^{d_1}\}\}$  be sets of additional attributes and methods of  $C^{d_1}$ .  $C^{d_1} = \langle Project(C^{s_1}, \{T_{a_-^{s_1}} : a_-^{s_1}\}, \{m_-^{s_1} : \{T_{p_-^{s_1}} : p_-^{s_1}\}\}), Extend(C^{s_1}, \{T_{a_+^{d_1}} : a_+^{d_1}\}, \{m_+^{d_1} : \{T_{p_+^{d_1}} : p_+^{d_1}\}\}) \rangle$ , where *Project* represents the Project operation, and *Extend* represents the Extend operation.

**Objects Associated with a Workflow Instance.** An object associated with a workflow instance need not be presented completely in a workflow view. The class that the original object belongs to is the source class, derived from which we define a deputy class whose objects are used in the workflow view. Some attributes are hidden from the view, which is realized via the project operation; some are made read-only via the switching operation of read and others are presented with write access via the switching operation of write. Input/output parameters may be the deputy classes of the class mentioned above.

**Definition 3.** Let  $C^{s_2} = \langle \{o^{s_2}\}, \{T_{a^{s_2}} : a^{s_2}\}, \{m^{s_2} : \{T_{p^{s_2}} : p^{s_2}\}\} \rangle$  be a source class corresponding to the class whose objects are associated with a workflow instance,  $\{T_{a_-^{s_2}} : a_-^{s_2}\}$  and  $\{m_-^{s_2} : \{T_{p_-^{s_2}} : p_-^{s_2}\}\}$  be subsets of attributes and methods of  $C^{s_2}$  which are allowed to be inherited by the class  $C^{d_2}$  that represents the class whose objects are used in workflow view,  $\{T_{a_+^{d_2}} : a_+^{d_2}\}$  and  $\{m_+^{d_2} : \{T_{p_+^{d_2}} : p_+^{d_2}\}\}$  be sets of additional attributes and methods of  $C^{d_2}$ .  $C^{d_2} = \langle Project(C^{s_2}, \{T_{a_-^{s_2}} : a_-^{s_2}\}, \{m_-^{s_2} : \{T_{p_-^{s_2}} : p_-^{s_2}\}\}), Extend(C^{s_2}, \{T_{a_+^{d_2}} : a_+^{d_2}\}, \{m_+^{d_2} : \{T_{p_+^{d_2}} : p_+^{d_2}\}\}) \rangle$ .

If the attribute  $a_-^{d_2}$  of  $C^{d_2}$  is inherited from the attribute  $a_-^{s_2}$  of  $C^{s_2}$ , and it can be read-only, the switching operation for inheriting  $a_-^{s_2}$  in form of  $a_-^{d_2}$  is realized in the following way:

$$read(o^{d_2}, a_-^{d_2}) \Rightarrow \uparrow f_{T_{a_-^{s_2}} \mapsto T_{a_-^{d_2}}} (read(o^{s_2}, a_-^{s_2})),$$

Here,  $\Rightarrow$  and  $\uparrow$  denote operation invoking and result returning, respectively.

If the attribute  $a_-^{d_2}$  of  $C^{d_2}$  is inherited from the attribute  $a_-^{s_2}$  of  $C^{s_2}$ , and it is presented with write access, the switching operation for inheriting  $a_-^{s_2}$  in form of  $a_-^{d_2}$  is realized in the following way:

$$write(o^{d_2}, a_-^{d_2}, v_-^{d_2}) \Rightarrow write(o^{s_2}, a_-^{s_2}, f_{T_{a_-^{d_2}} \mapsto T_{a_-^{s_2}}} (v_-^{d_2})).$$

**Events, Exceptions, Rules and Exception Handlers.** Events, exceptions and rules in the workflow view are defined as the deputy classes whose sources classes are the ones in the original workflow, which is realized by the project and extend operations. There definitions are same as Definition 3.

## 4.2 Realization of Different Workflow Views

For the taxonomy of workflow views discussed in Section 2.1, in this section we present the implementation strategies for three main kinds of workflow views.

The *Structural Transparency View* is used to determine whether the internal details of a workflow are visible by a user, the concept of which is identical to that of the process flow graph. Hence, it is also realized via the Project and Extend operation. The *Interaction view* is the interface of the workflow to an external user or agent to interact with the workflow itself. It is a subset of processes. This subset has interactions with the user or the agent. The original workflow class is the source class, whose deputy class is the workflow view. The process attributes in the workflow view are inherited from the original workflow, and those process attributes that are hidden are not, which is realized by the Project operation. The *Workflow Composition View* is a virtual workflow composed of components from different workflows. The original multi-workflows serve as the source classes, from which the composition workflow view is derived and defined as the deputy class via the join operation; the latter joins the processes of different workflows. Because of space limitations, we only give the formal definition of the composition workflow view in this paper.

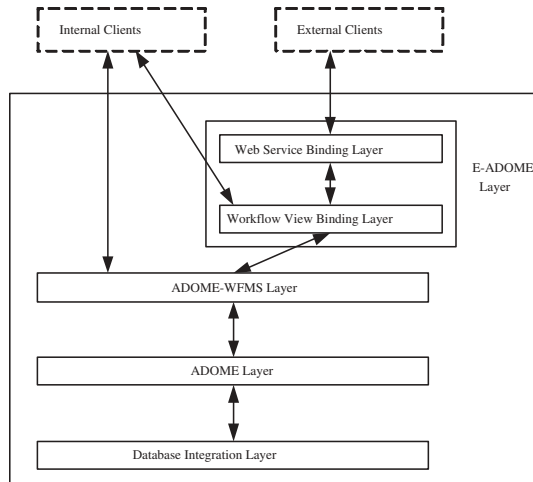
**Definition 4.** Let  $C_1^s = \langle \{o_1^s\}, \{T_{a_1^s} : a_1^s\}, \{m_1^s : \{T_{p_1^s} : p_1^s\}\} \rangle, \dots, C_n^s = \langle \{o_n^s\}, \{T_{a_n^s} : a_n^s\}, \{m_n^s : \{T_{p_n^s} : p_n^s\}\} \rangle$  be the source classes that represent a set of source workflows. A deputy class corresponding to the workflow composition view which is derived by the Join operation is represented as  $C^d = \text{Join}(C_1^s, \dots, C_n^s, cp)$ , where  $cp$  is a combination predicate. The extent of  $C^d$  is the set of deputy objects of aggregations of instances of  $C_1^s, \dots, C_n^s$  satisfying the combination predicate  $cp$ , which is expressed as  $\{o^d | o^d \rightarrow o_1^s \times \dots \times o_n^s, cp(o_1^s \times \dots \times o_n^s) == true\}$ .

## 5 Web Services Enactment

An advanced OO WFMS such as ADOME-WFMS [6] and its workflow view mechanism can be utilized to provide support for specifying, executing and monitoring composite web services, with necessary extensions. In this section, we describe E-ADOME as the extended framework for this purpose.

### 5.1 E-ADOME Architecture

The E-ADOME framework can be divided into the following layers: *Database Integration Layer* provides a uniform interface that applications can use to access and combine information from different system, in which the data in the local databases are all mapped into objects via an object-oriented interface. Details of this layer are described



**Fig. 4.** E-ADOME Architecture.

in [14]. *ADOME Layer* as an expert OODBMS [12] is employed to more adequately deal with data and knowledge management requirement of advanced information management applications, especially WFMSs. *ADOME-WFMS Layer* built upon ADOME facilities supports effective management of agents, on-line workflow evolution, automatic and cooperative exception handling [6]. *E-ADOME Layer* is the enhancement layer to the WFMS so as to enable ADOME-WFMS to interact with external web services through the Internet. The details of this layer are outlined in the subsection immediately below.

**E-ADOME Layer.** The E-ADOME layer allows an OO WFMS such as ADOME-WFMS to interact with external parties through web services. It includes two sub-layers, the *workflow view binding layer* and the *web services binding layer*.

1. The workflow view binding layer facilitates the definition, execution, interaction and management of workflow views. It supports internal clients to define different workflow views for specific requirements. Meanwhile, it also allows internal clients to interact with WFMS in a workflow view model (although such actions are supported by the organization model in most of the time). Also, the evolution of workflow views is supported by this layer.
2. The web services binding layer is in charge of the interactions with the web services environment. BPEL4WS [16] and ebXML BPSS [17] are two most popular web services technologies supporting business processes. For BPEL4WS, the web service binding layer can transform object-based workflow views in ADOME-WFMS to BPEL specifications, deploy the workflow view to BPEL engine and manage the exchange of instance data with BPEL engine. For ebXML BPSS, this layer maps workflow views to BPSS specifications, encapsulates the business process information into CPAs, and establishes the connection with external systems based on CPPs.



## 6 Related Work

There have some earlier works in the area of workflow views. Liu and Shen [13] presented an algorithm to construct a process view from a given workflow, but did not discuss its correctness with respect to inter-organizational workflows. Our preliminary approach of workflow views has been presented in [8]. From then, workflow views have been utilized as a beneficial approach to support the interactions of business processes in E-service environment [9][7][5]. However, most of these works focused on the conceptual level, and the implementation issues, especially those in the object-oriented environment, are largely neglected.

Van der Aalst and Kumar [2] presented an approach to workflow schema exchange in an XML dialect called XRL but it does not include the support for workflow views. Besides, van der Aalst [1] modeled inter-organizational workflows and the inter-organizational communication structures by means of Petri Nets and message sequence charts (MSCs), respectively. The soundness and consistency of the inter-organizational workflow can be analyzed by checking the consistency of Petri Nets against target MSCs. Since the author abstracted from data and external triggers, the proposed communication protocol is not as complete as the interoperation protocol presented in the workflow view approach [7]. To address the derivation of private workflows from inter-organizational workflows, Van der Aalst and Weske [3] used the concept of workflow projection inheritance introduced in [4]. A couple of derivation rules are proposed so that a derived workflow is behaviorally bisimilar to the original workflow based on branching semantics, in contrast to the trace semantics adopted in the workflow view model.

## 7 Conclusion

In this paper, we have adopted the object deputy model as the supporting mechanism for workflow views in an OO WFMS. Through its facilities, we have given the definitions of components of workflow views, i.e., flows, objects, exceptions, events, rules and exceptions handlers. Also three main kinds of workflow views, namely, structural transparency view, interaction view and composition view, are discussed for their implementation strategy. Based on a reference model of OO WFMS (viz. ADOME-WFMS), we put forward E-ADOME as an extended framework which supports the implements of workflow views in the web services environment.

Our future work will focus on the workflow view enactment in specific web services environments. For BPEL4WS, we will investigate the issues of date management between the instances of workflow view and BPEL. For ebXML BPSS, the business model based derivation of workflow views will be our main topic.

## Acknowledgement

The authors would like to thank Dr. Dickson K. W. Chiu for his beneficial comments on the construction of this paper.

## References

1. W.M.P. van der Aalst. Interorganizational workflows: An approach based on message sequence charts and petri nets. *Systems Analysis - Modelling - Simulation*, 34(3):335–367, 1999.
2. W.M.P. van der Aalst and A. Kumar. Xml based schema definition for support of inter-organizational workflow. *Information Systems Research*, 14(1):23–46, 2003.
3. W.M.P. van der Aalst and M. Weske. The p2p approach to interorganizational workflows. In *13th International Conference Advanced Information Systems Engineering (CAiSE 2001)*, volume 2068 of *Springer LNCS*, pages 140–156, Interlaken, Switzerland, 2001.
4. T. Basten and W.M.P. van der Aalst. Inheritance of behavior. *Journal of Logic and Algebraic Programming*, 47:47–145, 2001.
5. S. C. Cheung, D. K. W. Chiu, and S. Till. A three-layer framework for cross-organizational e-contract enactment. *Web Services, E-Business, and the Semantic Web*, 2512:78–92, 2002.
6. Dickson K. W. Chiu, Qing Li, and Kamalakkar Karlapalem. Adome-wfms: towards cooperative handling of workflow exceptions. In *Advances in exception handling techniques*, pages 271–288. Springer-Verlag New York, Inc., 2001.
7. Dickson K.W. Chiu, S.C. Cheung, Sven Till, Kamalakkar Karlapalem, Qing Li, and Eleanna Kafeza. Workflow view driven cross-organizational interoperability in a web service environment. *Information Technology and Management*, to appear, 2004.
8. Dickson K.W. Chiu, Kamalakkar Karlapalem, and Qing Li. Views for inter-organization workflow in an e-commerce environment. In *Semantic Issues in E-Commerce Systems, IFIP TC2/WG2.6 Ninth Working Conference on Database Semantics*, Hong Kong, 2001. Kluwer.
9. Dickson K.W. Chiu, Kamalakkar Karlapalem, Qing Li, and Eleanna Kafeza. Workflow view based e-contracts in a cross-organizational e-services environment. *Distributed and Parallel Databases*, 12(2-3):193–216, 2002.
10. Yahiko Kambayashi and Zhiyong Peng. An object deputy model for realization of flexible and powerful objectbases. *Journal of Systems Integration*, 6:329–362, 1996.
11. Gerti Kappel, Stefan Rausch-Schott, and Werner Retschitzegger. A framework for workflow management systems based on objects, rules and roles. *ACM Comput. Surv.*, 32:27, 2000.
12. Q. Li and F.H. Lochovsky. Adome: an advanced object modeling environment. *Knowledge and Data Engineering, IEEE Transactions on*, 10(2):255–276, 1998.
13. Duen-Ren Liu and Minxin Shen. Modeling workflows with a process-view approach. In *Database Systems for Advanced Applications, 2001. Proceedings. Seventh International Conference on*, pages 260–267, 2001.
14. Yi Luo, Zhiyong Peng, Yi Guo, Yixian Liu, and Xun Song. Integration of multiple heterogeneous databases in smalltalk. In *CNCC*, 2003.
15. OMG. <http://www.omg.org/>.
16. M. Snoeck, S. Poelmans, and G. Dedene. An architecture for bridging oo and business process modelling. In *Technology of Object-Oriented Languages, 2000. TOOLS 33. Proceedings. 33rd International Conference on*, pages 132–143, 2000.
17. Gottfried Vossen and Mathias Weske. The wasa2 object-oriented workflow management system. In *Proceedings of the 1999 ACM SIGMOD international conference on Management of data*, pages 587–589, Philadelphia, Pennsylvania, United States, 1999. ACM Press.
18. G. Wirtz, M. Weske, and H. Giese. The ocon approach to workflow modeling in object-oriented systems. *Information Systems Frontiers*, 3(3):1387–3326, 2001.

# Modeling QoS for Semantic Equivalent Web Services\*

Derong Shen, Ge Yu, Tiezheng Nie, Rui Li, and Xiaochun Yang

Northeastern University, Shenyang, P. R. China  
{shendr, yuge}@mail.neu.edu.cn

**Abstract.** Quality of Service (QoS) is an important factor for selecting a better Web service from numerous semantic equivalent Web services in Web services composition. In this paper, a QoS model (e\_QoS) for evaluating semantic equivalent Web services is proposed. In which, the factors used to evaluate Web Services include Accessed Times, Invoked Success Rate, Average Responding Time, Stability Variance, and Customers Estimation. The performance of Web services in Latest Period of Time, the performance in history and the performance estimated by customers are considered together. By experiments, we concluded that e\_QoS is effective for estimating Web services, and can improve the quality of web services composition greatly.

## 1 Introduction

Semantic equivalent Web services are a group of Web services, which can replace each other, if a Web service is invalid at runtime of a composite service, another semantic equivalent Web service can be selected to replace it and make the composite Web service running on. But, with the booming development of Web services and the increased available Web services on the Internet, it is becoming challengeable to discover a suitable web service from numerous semantic equivalent Web Services, so as to provide a composite services with better performance for customers. There are many research papers about the discovery of Web services [1],[2],[3], these researches mainly addressed where the Web services to be located without discussing the performance of Web services. Many technologies, such as WSDL, UDDI, SOAP, WSFL[4], XLANG[5], BPEL4WS[6] and DAML-s[7], have been proposed for supporting Web services, but only DAML-S mentioned the QoS for Web services. In DAML-S, QoS attributes as non-function attributes are quantificationally given by services providers without specifications followed. Currently, study on QoS for Web services includes the performance evaluation of Web services and the performance evaluation of Web services composition.

This paper focused on the QoS for Web services and proposed a QoS model named e\_QoS that can be applied to discover Web services and make Web services composition more reliable and efficient.

---

\* This research is supported by the National High-Tech Development Program (2003AA414210), the National Science Foundation (60173051).

## 2 Related Work

Some Related researches [8–13] proposed the idea about evaluating the QoS for Web services and emphasized the importance of QoS. For example, Jorge Cardoso instructionally proposed the factors used in a comprehensive QoS model [8], which include cost, time, reliability and fidelity. In [9], Chandrasekaran proposed that the specification of Web services should include syntactic, semantic and QoS, and the research main pays attention to time attribute. In [10], Sumra R suggested that the evaluating factors of QoS for Web services should include performance, reliability, integrity, accessibility, availability, interoperability and security, and in which, the performance factor consists of throughput, latency time, execution time and transaction time. The QoS template was presented in [11], which included cost, time and company, and the Web services are discovered based on the similarity between the quality information described by the service providers and required by service consumers. In [12], the quality model for the workflows based on Web services and the factors including time, cost, reliability and fidelity were introduced in details, the factors of evaluation in the QoS model absolutely depend on the historical statistic information, and the simulating result of the model and the practical result had been actually compared. In [13], additional information about the component quality factors is given by providers and used for querying QoS-enabled application using UDDI database.

In e\_QoS, the information with objectivity recorded in the process of invoking Web services as the main information is used for evaluating the performance of Web services, and the final performance of Web services evaluated in e\_QoS is composed of the performance in the Latest Period of Time, the performance in history and the performance estimated by customers. It can reflect the performance of Web service objectively.

## 3 e\_QoS

In e\_QoS, we take the following considerations: (1) e\_QoS must be of adaptable automatically. (2) The performance of Web services in Latest Period of Time is extremely important, even exceed the performance evaluated based on the historical information. (3) The information recorded by central Executing Agent [14] should be play an important role for evaluating the QoS of Web services. (4) Factors set in e\_QoS mainly include Accessed Times, Invoked Success Rate, Average Responding Time, Stability Variance, and Customers Estimation. (5) Weights can be set for factors based customer's requirements to represent the personality of e\_QoS.

With the ideas discussed above, e\_QoS is proposed, in which, the performance of Web services in Latest Period of Time, the performance in history and the performance estimated by consumers are considered together. To quantify the quality of Web services, each evaluating result is denoted as a score in range of 0 and 1, and the total score computed according to a given evaluating formula is the final evaluating result, and the Web services with higher score is better.

First, we'll give some related concepts before e\_QoS is addressed in detail. Log Information (LogInfo) is the information that is recorded by Executing Agent in the execution of Web services, including the Web services unique identity (serviceID), the executing state of Web services, the start-time and the end-time of the execution of web services. Checkpoint is the time points designated periodically. At Checkpoint, LogInfo recorded during the period of time from previous Checkpoint to the current Checkpoint has been summed, and the summed information at the Checkpoint is called the CheckpointInfo, the period of time is called Latest Period of Time. Latest-Checkpoint is the latest Checkpoint, and the CheckpointInfo collected at Latest-Checkpoint is called Latest CheckpointInfo. HistoryInfo is the summed information collected based on all the LogInfo, or it is the sum of all the CheckpointInfo. Accessed Times is the times that a Web service was accessed. Invoked Success Times is the times that a Web service was invoked successfully. Invoked Success Times divided by Accessed Times is Invoked Success Rate. Responding Time is an interval from the start time of a Web service invoked to the time responding results returned, and Average Responding Time is the average Responding Time of a Web service.

In e\_QoS, the performance of Web services in Latest Period of Time is quantitatively denoted as the score of LatestCheckpoint, and the performance of Web services in history is quantitatively denoted as the historical score. In the following, we will give the detailed explanation about the effect of factors in e\_QoS.

### 3.1 Score of LatestCheckpoint ( $M_c$ )

For simplicity, based on Latest Checkpointinfo, the Accessed Times is denoted as  $N_c$ , the Invoked Success Rate as  $R_c$ , the Invoked Success Times as  $N_{cs}$ , the Average Responding Time as  $\overline{T_c}$ , and Stability Variance is denoted as  $S_c$ .

**Definition 1 (Stability Variance ( $S_c$ )).** It describes the stability of Web services by comparing the Responding Time of a Web service in different invoked events in Latest period of Time. The Web services with less difference of Responding Time are

$$S_c = \sqrt{\frac{\sum_{i=1}^n T_i^2}{n} - \overline{T_c}^2} \quad (1)$$

more stable. By modifying the mathematical variance, it is denoted as follows: where,  $T_i$  is the Responding Time of a Web service in a invoked event in Latest period of Time,  $n$  is the  $N_{cs}$  of the Web service.

**Definition 2 (The corresponding score of  $N_c$  ( $M_{cl}$ )).** It is used to represent the reliability of Web services with  $R_c$ , and denoted as follows:

$$M_{c1} = \begin{cases} 1, & N_c > p_1 \\ N_c / p_1, & N_c \leq p_1 \end{cases} \quad (2)$$

where,  $P_1$  is a threshold value, see Section 3.4.

**Definition 3 (The corresponding score of  $R_c$  ( $M_{c2}$ )).** It is a major factor for representing the reliability of Web services, and denoted as follows:

$$M_{c2} = N_{cs} / N_c \quad (3)$$

**Definition 4 (The corresponding score of  $\overline{T_c}$  ( $M_{c3}$ )).**  $M_{c3}$  is denoted as follows: where,  $P_{c2}$  is a threshold value, see Section 3.4.

$$M_{c3} = \begin{cases} 1, & \overline{T_c} \leq p_{c2} \\ e^{(1 - \overline{T_c} / p_{c2})}, & \overline{T_c} > p_{c2} \end{cases} \quad (4)$$

**Definition 5 (The corresponding score of  $S_c$  ( $M_{c4}$ )).**  $M_{c4}$  is denoted as follows:

$$M_{c4} = \begin{cases} 1, & S_c \leq p_3 \\ e^{1 - S_c / p_3}, & S_c > p_3 \end{cases} \quad (5)$$

where,  $P_3$  is a threshold value, see Section 3.4.

So, aggregating the  $M_{c1}$ ,  $M_{c2}$ ,  $M_{c3}$ ,  $M_{c4}$ , and the formula of total score at Latest-Checkpoint is:

$$M_c = \sum_{i=1}^n (w_{ai} * M_{ci}) \quad (6)$$

where,  $n=4$ ,  $w_{a1}$ ,  $w_{a2}$ ,  $w_{a3}$ ,  $w_{a4}$  are the weights of  $M_{c1}$ ,  $M_{c2}$ ,  $M_{c3}$ ,  $M_{c4}$ .

### 3.2 The Historical Score ( $M_l$ )

It plays a secondary important role in e\_QoS, the factors set used to evaluate the Web services include Accessed Times Invoked Success Rate and Average Responding Time. For simplicity, based on HistoryInfo, the Accessed Times is denoted as  $N_p$ , the Invoked Success Times is denoted as  $N_{ts}$ , the Invoked Success Rate is denoted as  $R_p$ , and the Average Responding Time is denoted as  $\overline{T_i}$ .

In order to make  $M_{li}$  reasonable, multi-segments of Accessed Times are set and adjusted by the experienced system administrator based on the status of system running. Suppose three segments are:  $N_L$ ,  $N_M$  and  $N_H$ , and their regions of relative score are  $0 \sim y_0$ ,  $y_0 \sim y_1$  and  $y_1 \sim y_2$ , now let both of slopes in the region of  $[0, N_L]$  and  $[N_M, N_H]$  be lower than that in  $[N_L, N_M]$ , then the Accessed Times in the region of  $[N_L, N_M]$  is more important than that in other regions.

**Definition 6 (The corresponding score of  $N_t$  ( $M_{t1}$ )).** Suppose  $x$  is  $N_t$  of a Web service, and then  $M_{t1}$  is denoted as follows:

$$M_{t1}=y=\begin{cases} (y_0/N_L) x, & x \in [0 \sim N_L], y \in [0, y_0) \\ [ (y_1 - y_0)/(N_M - N_L)]x + \{y_0 - [ (y_1 - y_0)/(N_M - N_L)] * N_L\}, & x \in [N_L \sim N_M], y \in [y_0, y_1) \\ [ (y_2 - y_1)/(N_H - N_M)]x + \{y_1 - [ (y_2 - y_1)/(N_H - N_M)] * N_M\}, & x \in [N_M \sim N_H], y \in [y_1, y_2] \end{cases} \quad (7)$$

**Definition 7 (The corresponding score of  $R_t$  ( $M_{t2}$ )).**  $M_{t2}$  is denoted as follows:

$$M_{t2} = N_{ts}/N_t \quad (8)$$

**Definition 8 (The corresponding score of  $\bar{T}_t$  ( $M_{t3}$ )).**  $M_{t3}$  is denoted as follows:

$$M_{t3} = \begin{cases} 1, & \bar{T}_t \leq p_{t2} \\ e^{(1-\bar{T}_t)/p_{t2}}, & \bar{T}_t > p_{t2} \end{cases} \quad (9)$$

where,  $p_{t2}$  is a threshold value, see Section 3.4.

Hence, the formula of the historical score ( $M_t$ ) is denoted as follows:

$$M_t = \sum_{i=1}^{n-1} (w_{ai} * M_{ti}) * \sum_{i=1}^n w_{ai} / \sum_{i=1}^{n-1} w_{ai} \quad (10)$$

where,  $n=4$ , and  $w_{a1}$ ,  $w_{a2}$ ,  $w_{a3}$  are the weights of  $M_{t1}$ ,  $M_{t2}$ ,  $M_{t3}$  respectively.

### 3.3 The Score of Consumers Estimation ( $M_u$ )

It represents the evaluation of customers on Web services invoked by them, and plays an additional role.  $M_u$  is denoted as follows:

$$M_u = \begin{cases} 0, & count < LCount \\ avg + adds, & count \geq LCount \end{cases} \quad (11)$$

$$adds = \begin{cases} (avg / H_{score}) * (count / HCount), & count < HCount \\ (avg / H_{score}), & count \geq HCount \end{cases} \quad (12)$$

where,  $count$  is the number of consumers' votes,  $avg$  is the average score of Consumers Estimation,  $LCount$  is the lower threshold value set based on the types of semantic equivalent Web services,  $H_{score}$  is the highest score designated,  $HCount$  is the upper threshold value.

In order to limit the value of  $adds$  in range of 0 to 1, the formula of  $M_u$  is modified as follows:

$$M_u = (avg + adds) / (H_{score} + 1) \quad (13)$$

At last, let  $M_1=M_c$ ,  $M_2=M_p$ ,  $M_3=M_u$ , the final score of a Web service evaluated by means of e\_QoS model is computed by the following formula:

$$TOTLE = \sum_{i=1}^m (w_{hi} * M_i) \quad (14)$$

where, let  $m=3$ , and  $w_{b1}$ ,  $w_{b2}$ ,  $w_{b3}$  are the weights of  $M_1$ ,  $M_2$ ,  $M_3$  respectively.

### 3.4 Threshold Values

The rule that called bis-average is used for calculating threshold values. Based on the simple but very efficient rule, these threshold values ( $p_1, p_{c2}, p_{t2}, p_3$ ) were assigned as follows:

$$\begin{aligned} p_1 &= (N_{AVG} + N_{MAX}) / 2 \\ p_2(p_{c2}, p_{t2}) &= (T_{AVG} + T_{MIN}) / 2 \\ p_3 &= (S_{AVG} + S_{MIN}) / 2 \end{aligned} \quad (15)$$

where,  $N_{AVG}$  are the average  $N_c$  of all semantic equivalent Web services, and  $N_{MAX}$  is the maximum of  $N_c$ .  $T_{AVG}$  are the average  $\overline{T_c \overline{T_i}}$  of all semantic equivalent Web services, and  $T_{MIN}$  is the minimum of  $\overline{T_c \overline{T_i}}$ .  $S_{AVG}$  are the average  $S_c$  of all semantic equivalent Web services, and  $S_{MIN}$  is the minimum of  $S_c$ .

## 4 Experiments

The experiments are run in our campus network with 100M-switched Ethernet, a PC Server of 0.7G CPU and 2GB RAM, and several PC Client of 2.4GB CPU, 0.5GB RAM. The operation systems of Windows Server 2003 and Windows 2000 are used respectively, and the e\_QoS package is developed by Microsoft C#. e\_QoS runs in the Server to sum the CheckpointInfo and HistoryInfo based on LogInfo, and these statistic information is used to calculate the total score of semantic equivalent Web services by Discovering Agent[14] in the execution of composite services. Web services run in the Client computers. The test results address on two aspects: (1) The characteristics of e\_QoS itself. (2) The effect on Web services composition.

### 4.1 Simulation Measures and Experiment Test

In the following, a composite service is abbreviated as C-WS. To simulate the real execution status of composite services in Internet environment, some measures are adopted, such as invalid Web services, selection of Web services, the execution of the composite services with invalid Web services, etc.

**Setting Invalid Web Services and Timeout Cost:** In the execution of composite Web services, invalid Web services will be set to simulate the real running process of composite services in Internet environment by means of Invoked Success Rate ( $R_c$ ), it is denoted as:

$$f(WS_i) = \begin{cases} 1 & \text{Random}(1,1.0) \geq R_c \\ 0 & \text{Random}(1,1.0) < R_c \end{cases} \quad (16)$$



where,  $f(WS_i)$  represents the state of  $WS_i$  invoked, “0” represents the  $WS_i$  is invoked successfully, otherwise, the value of  $f(WS_i)$  is “1”.  $Random(1,1.0)$  represents a float value in the range of values of 0 and 1.0 created at random.

If  $f(WS_i)=1$ , a timeout ( $T_o(WS_i)$ ) for waiting failure information of  $WS_i$  is set, it is denoted as:

$$T_o(WS_i)=3*\overline{T_i} \quad (17)$$

where,  $\overline{T_i}$  represents the  $\overline{T_c}$  of  $WS_i$ .

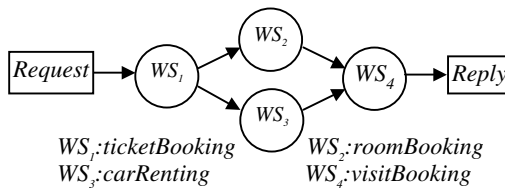
**Methods for Selecting Web Services:** In C-WS without e\_QoS, Web services are selected at random, and In C-WS with e\_QoS, Web services are selected according to the quality of service evaluated based on e\_QoS, and we give the Web services with higher score priority.

**Setting Responding Time and Failure Rate of C-WS:** In order to explain the effect of e\_QoS, the average responding time and the failure rate of both C-WSs without e\_QoS and with e\_QoS are compared, in which, the failure rate of C-WS is denoted as  $R_{c-WS}$ , and the average responding time of C-WS is denoted as  $\overline{T_{c-WS}}$ .  $R_{c-WS}$  is represented as:

$$R_{c-WS} = \frac{\sum_{i=1}^n \sum_{j=1}^m f(WS_j)}{(\sum_{i=1}^n \sum_{j=1}^m f(WS_j) + n * m)} \quad (18)$$

where,  $m$  is the number of Web services included in C-WS, and  $n$  is the total times of C-WS executed.

For  $\overline{T_{c-WS}}$ , it has different cost for different samples of composite services. For example, in Fig. 1, where, each Web service has lots of semantic equivalent Web services, when a request comes up, Travel Service will be executed. When a Web service that will be invoked is invalid, another semantic equivalent Web services must be selected and invoked, such that the execution of a composite Web service can continue till to its end. The Responding Time of C-WS without invalid Web services is:  $\overline{T_{c-WS}}=T_1+[T_2/T_3]+T_4$ , where,  $T_1, T_2, T_3, T_4$  are the Responding Time of  $WS_1, WS_2, WS_3$  and  $WS_4$  respectively,  $[T_2/T_3]$  represents that the longer responding time are chosen. In fact,  $\overline{T_{c-WS}}$  is composed of  $\overline{T_{c-WS}}$  without invalid Web services and all the  $T_o(WS_i)$ .



**Fig. 1.** The Test Sample: Travel Service

## 4.2 Test Results

Two aspects of experiments are discussed: Firstly, a group of semantic equivalent Web services are evaluated to explain e\_QoS efficiently. Secondly, e\_QoS is applied in a sample scenario discussed in Fig 1, and the effect of e\_QoS on the execution of the composite service is given. In the experiments, suppose, the parameters are configured as: (1)  $H_{score}=1$ ,  $LCount=20$ ,  $HCount=100$ ,  $N_L=100$ ,  $N_M=1000$ ,  $N_H=10000$ ,  $y_0=0$ ,  $y_1=1/4$ ,  $y_2=3/4$ . (2)  $W_{b1}=0.6$ ,  $W_{b2}=0.2$ ,  $W_{b3}=0.2$ , since  $M_c$  is the most important among  $M_c$ ,  $M_t$  and  $M_u$  in e\_QoS. However, the values can be configured on demand.

### (1) The Test Results with a Group of Semantic Equivalent Web Services

We choose ten Web services which are semantic equivalent to  $WS_2$  (see Fig. 1) to be addressed in the experiment, in which, the ten Web services are deployed in Clients respectively and invoked thousands of times, then the Latest CheckpointInfo and HistoryInfo are obtained. Based on these collected information, the Scores of factors are calculated, which are shown from Fig.2 to Fig.6, where *UUID* represents Web Service unique identity. The explanation of the test results is as follows:

- Scores of factors with equal weights: The scores of factors at LatestCheckpoint are shown in Fig. 2, the scores of factors in history are shown in Fig.3, the scores of factors for the total scores are shown in Fig.4, where, weights are set as:  $W_{a1}=W_{a2}=W_{a3}=W_{a4}=0.25$ .
- Scores of factors with varied weights: The diagram in Fig.5 and Fig.6 show the evaluation results of Web services when different factors are emphasized, in which, besides the weights of factors emphasized is assigned to 0.7, others are all assigned to 0.1. Here, the Web service with *UUID*=2 is as an example, in Fig.2, we learn that  $M_{c2}$  and  $M_{c3}$  are higher than  $M_c$ ,  $M_{c1}$  and  $M_{c4}$  are lower than  $M_c$ , so, the score with equal weights should lower than both the scores emphasizing  $R_c$  and  $T_c$ , and higher than both the scores emphasizing  $N_c$  and  $S_c$ . The test results shown in Fig.5 are conforming to results analyzed in theory above. The scores in history are similar to the results at LatestCheckpoint.

### (2) The Test Results with Composite Services

Based on the Sample: Travel Service shown in Fig.1, four groups ( $WS_1$ ,  $WS_2$ ,  $WS_3$  and  $WS_4$ ) of Latest CheckpointInfo and HistoryInfo are obtained by Travel Service and other composite services running thousands of times. Then, both Travel Service with e\_QoS (abbreviated TravelServiceQoS ) and Travel Service without e\_QoS (abbreviated TravelServiceNonQoS) are executed.

Here only two groups of test results are given, which are shown in Fig. 7 and Fig. 8, in which, NonQoS represents that e\_QoS is not applied in Travel Service, *QoS-E* represents e\_QoS with equal weights, and *QoS-Nc-Rc* represents e\_QoS with  $R_c$  and  $N_c$  emphasized, *QoS-Tc-Sc* represents e\_QoS with  $T_c$  and  $S_c$  emphasized. We have learned:

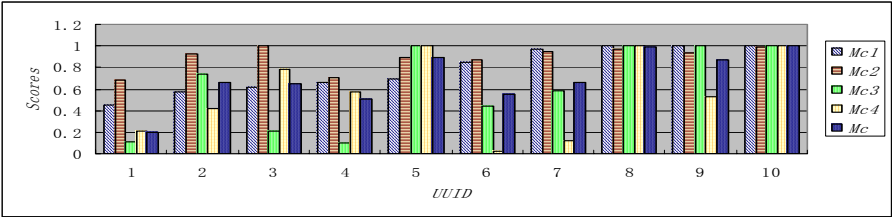


Fig. 2. Scores of factors at LatestCheckpoint

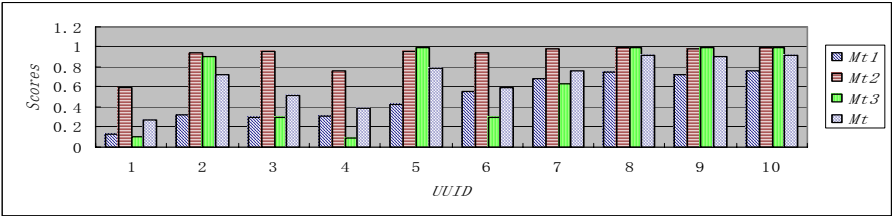


Fig. 3. Scores of factors in history

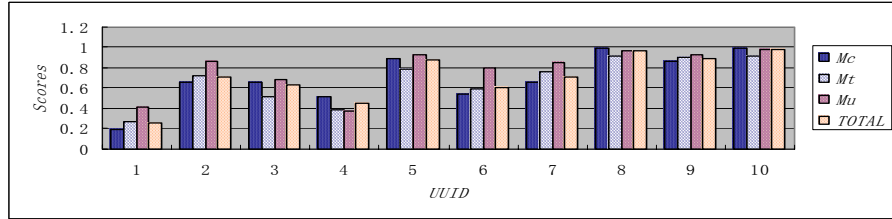


Fig. 4. Total scores

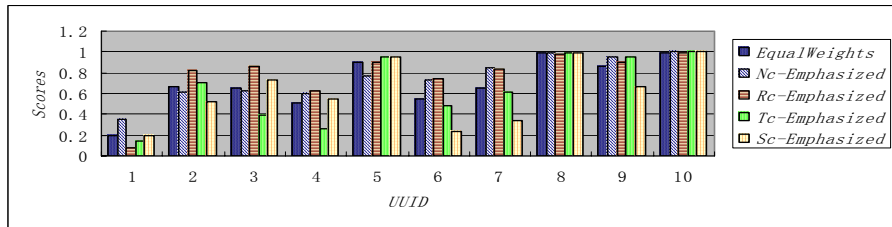
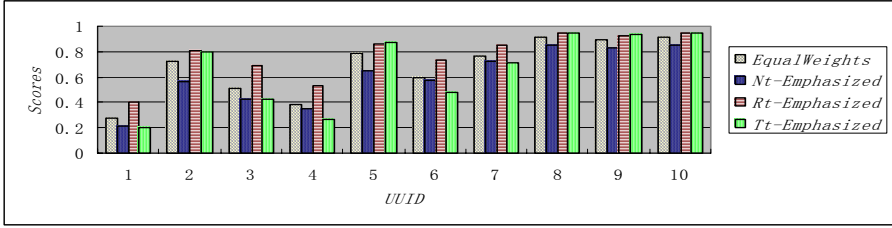
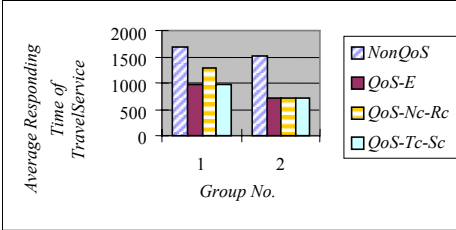


Fig. 5. Scores impacted by weights at LatestCheckpoint

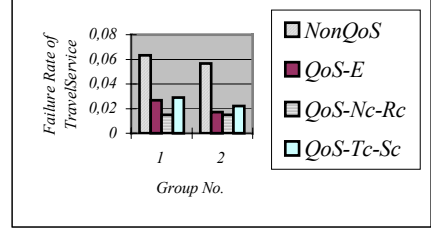
- The average responding time of TravelServiceNonQoS is longer than that of TravelServiceQoS and the failure rate of TravelServiceNonQoS is higher than that of TravelServiceQoS.
- The average failure rate of TravelServiceQoS with  $R_c$  and  $N_c$  emphasized are the smallest one, but the average responding time of it may be longer.
- The average responding time of TravelServiceQoS with  $T_c$  and  $S_c$  emphasized maybe not the shortest one due to the influence of failure rate.



**Fig. 6.** Scores impacted by weights in history



**Fig. 7.** Average responding time of TravelService



**Fig. 8.** Failure rate of TravelService

- TravelServiceQoS with equal weights are better in both the average failure rate and the average responding time.
- If we have a stable and better Web running environment, and Web services are available all the time, only  $T_c$  emphasized will has a good answer, otherwise, the equal weights should be adopted.

Based on the test results, the following conclusions are induced:

- The autonomous, dynamic characteristics of Web services can be embodied in e\_QoS efficiently.
- e\_QoS satisfies the requirements of various customers.
- The composite services with e\_QoS make the execution of composite services more efficient and reliable.
- e\_QoS can improve the quality of Web services composition efficiently.
- The model may be helpful and useful to the related researches on the QoS evaluation.

## 5 Conclusions

Due to plenty of existing autonomic and heterogeneous Web services, we proposed e\_QoS for evaluating the QoS of semantic equivalent Web services. The experiments verify that e\_QoS reflects the characteristics of Web services objectively, and make Web service composition more strong and efficient. The advantages of e\_QoS model manifest in the following aspects:

- It emphasizes the characteristics that the performance of Web services in the Latest Period of Time is more important.
- It supports the personalized requirement of customers
- It has self-adaptability and scalability.
- It is suitable to the autonomy, dynamic characteristics of Web services.

## References

1. Paolucci M., Kawamura T.: Importing the Semantic Web in UDDI. Proceedings Web Services, E-Business and Semantic Web Workshop, CAiSE (2002), 225-236
2. ESPEAK.: Hewlett Packard's Service Framework Specification. HP Inc, 2000
3. Bernstein A.: Discovering services: Towards High-Precision Service Retrieval in Proceedings of the CaiSE workshop on Web Services (2002).
4. Leymann F.: Web Services Flow Language (WSFL 1.0). <http://4.ibm.com/software/solutions/webservices/pdf/WSFL.pdf> (2001)
5. Thatte S.:XLANG: Wsb Services for Business Process Design. [http://www.gotdot.net.com/team/xml\\_wsspecs/xlang-c/default.htm](http://www.gotdot.net.com/team/xml_wsspecs/xlang-c/default.htm), Microsoft Corporation (2001)
6. Curbera F., Golland Y.: Business process execution language for Web Services, Version 1.0. <http://www.ibm.com/developerworks/library/ws-bpel/> (2002)
7. Burstein M., Hobbs J., Lassila O.: DAML-S: semantic markup for Web services. Proceedings of the International Semantic Web Workshop (2001)
8. Cardoso J., Bussler C.: Semantic Web Services and Processes: Semantic Composition and Quality of Service. On the Move to Meaningful Internet Computing and Ubiquitous Computer 2002, Irvine CA (2002).
9. Chandrasekaran S., Silver G.: Web service technologies and their synergy with simulation. In Proc. of the 2002 Winter Simulation Conference (WSC'02), (2002)
10. Sumra R., Arulazi D.: Quality of Service for Web Services-Demystification, Limitation, and Best Practices. <http://www.developer.com/services> (2003)
11. Fengjin W., Zhoufeng Z.: A Dynamic Matching and Binding Mechanism for Business Services Integration. In Proc. of the EDCIS 2002, September (2002) 17-20
12. Cardoso J., Miller J.: Modeling Quality of Service for Workflows and Web Service Processes. The International Journal on Very Large Data Bases (VLDBJ), Verlag, Berlin-Heidelberg (2003)
13. Farkas P., Charaf H.: Web Services Planning Concepts. WSCG'2003, Plzen, Czech Republic (2003)
14. WU Q., Shen D., Yu G.: A Web Service Composition System for Dynamic Alliances, Computer Integrated Manufacturing Systems, Vol.9 (2003) 674-679 (in Chinese)

# Improved Email Classification through Enriched Feature Space\*

Yunming Ye<sup>1</sup>, Fanyuan Ma<sup>1</sup>, Hongqiang Rong<sup>2</sup>, and Joshua Zhexue Huang<sup>2</sup>

<sup>1</sup> Department of Computer Science and Engineering, Shanghai Jiao Tong University,  
Shanghai 20030, China

{yym,fyma}@sjtu.edu.cn

<sup>2</sup> E-Business Technology Institute, The University of Hong Kong, Hong Kong, China  
{hrong,jhuang}@eti.hku.hk

**Abstract.** This paper presents a novel feature space enriching (FSE) technique to address the problem of sparse and noisy feature space in email classification. The (FSE) technique employs two semantic knowledge bases to enrich the original sparse feature space, which results in more semantic-rich features. From the enriched feature space, the classification algorithms can learn improved classifiers. Naive Bayes and support vector machine are selected as the classification algorithms. Experiments on an enterprise email dataset have shown that the FSE technique is effective for improving the email classification performance.

**Keywords:** Email Classification, Feature Space Enriching, Semantic Knowledge Base, Text Categorization

## 1 Introduction

With the explosive growth of emails in daily life, how to process and manage the email flow efficiently presents a big challenge, especially for large organizations such as e-government and large-scale companies. As a result, some intelligent email classification (EC) approaches have been explored to attack this problem [1–3], which automatically classify emails into predefined categories and thus greatly facilitate the management of emails.

In general framework, EC can be viewed as a special case of text classification, but the problem of sparse and noisy feature space makes it more challenging. Compared to general natural language text, email messages are often of short length, and they are noisier due to many informal words and sentences, such as unusual abbreviation, spelling errors etc. Hence the feature space for EC is always far sparser, which presents a difficulty for accurate and stable classification. Moreover, the training samples for EC are usually not in large batch but in dynamic flow with possibly changing content, so the training collection may not provide enough information for the classification algorithms to learn an accurate classifier. While previous work on EC mainly focused on the performance

---

\* This paper has been supported by China NSF project (No.60221120145) and Shanghai Science and Technology Foundation (under the granted project No.02DJ14045).

comparison of different classification algorithms [1–7], they seldom addressed the above special characteristics for EC. We argue that this is the primary limitation for the further success of current EC approaches. To improve the performance of EC, these special characteristics should be exploited carefully.

In this paper, we propose a novel feature space enriching (FSE) technique to address the above problems of EC. In the process of FSE, semantic features that are related to the terms in the original feature space, are extracted from the concept hierarchies of two semantic knowledge bases (WordNet and HowNet). Then the final enriching features are selected from the semantic features based on the ‘information gain increment’ filtering metric, and used to enrich the original sparse feature space. The resulting feature spaces will provide semantic-rich features for classification algorithms to learn improved classifiers. Naive Bayes and support vector machine (SVM) are selected as the classification algorithms, which learn classifiers with more accuracy and greater generalization ability from the enriched feature space. Comprehensive experiments on a real-world bilingual (Chinese-English) enterprise email dataset show that the FSE technique is effective for improving the email classification accuracy.

The rest of this paper is organized as follows. In section 2, we discuss some related work. Section 3 describes the details of the FSE technique based on WordNet and HowNet. Experiment results and analysis are presented in section 4. Section 5 concludes and points out some lines of future work.

## 2 Related Work

In [1], Cohen used the rule learning algorithm RIPPER to induce sets of keyword-spotting rules that compose the classifiers for classifying email. In the experiments, he compared his method to TFIDF method and got similar accuracy. [3] by Provost also employed RIPPER algorithm for email classification and compared it to Naive Bayes(NB) classifier. All the experiments showed that NB outperformed RIPPER in accuracy. In [2], an ILP framework was used to enable composite rule learner that combined explicit hypothesis construction approach with instance based learning approach. They found the composite rule classifier had the best precision on the items that were classifiable by it.

TFIDF in IR community is another approach that has been extensively used. MailCat [6] employed a TFIDF approach which computed weighted vectors for each class based on word frequencies, then the weighted vectors were used to classify each new message based on the cosine distance. Its improved version SwiftFile[7] addressed the importance of incremental learning for email classification and the experiments showed that incremental learning gained higher accuracy than batch learning. Naive Bayes is another frequently used approach for email classification and anti-spam, and often acts as baseline classifier for evaluating other approaches. The iFile [4] by Rennie is a filter for EXMH mail client based on naive Bayes classification. The experiments on a number of iFile users have shown high performance. Other methods used include decision tree by Diao et al [5], and SVM by Brutlag et al [8].

Most of previous work emphasized on comparing performance of different classification algorithms, and seldom address the special characteristics of email domain as mentioned in section one. Also they mainly focused on personal email classification and seldom addressed the email classification for large enterprises or organizations. These issues were explored in this paper.

### 3 Feature Space Enriching for Email Classification

#### 3.1 Formal Representation of Email Dataset

Like text classification, the first step for email classification is to transform email content into a formal representation suitable for classification algorithms. Traditionally, bag-of-words representation is employed for this task, where an email is represented as a set of words (or terms) and word position is ignored. Thus, the feature space for email classification can be represented as a list of terms:

$$\vec{V}_o = \{t_1, t_2, \dots, t_k, \dots, t_n\} \quad (1)$$

where  $t_k$  is a term that occurs in the email training set  $\mathbf{T}$ ,  $n$  is the total vocabulary size of  $\mathbf{T}$ .

Terms differs in their discrimination ability for classification of emails: some are critical for deciding the categories of emails, while others are just inessential. Therefore a proper term weighting scheme is required to encode this difference. In our work, we use a modified vector space model (VSM) [9] from information retrieval community to represent email content. Each email  $m_i$  is represented as a weighed term vector as follows:

$$m_i = \{\omega_{1i}, \omega_{2i}, \dots, \omega_{ki}, \dots, \omega_{ni}\} \quad (2)$$

where  $n$  is the total vocabulary size of the email training set  $\mathbf{T}$ ,  $\omega_{ki}$  is the weight of term  $t_k$  in the email training set and is calculated by  $TF*IDF$  weighting scheme:

$$\omega_{ki} = TF_{ki} * IDF_{ki} \quad (3)$$

where  $TF_{ki}$  is the frequency of term  $t_k$  in email  $m_i$ .  $IDF_{ki}$  is the inverse class frequency which makes the bias that terms appear in many categories are less useful for classification. It is computed as follows:

$$IDF_{ki} = IDF_{kc_j} = \log\left(\frac{N}{n_k}\right) \quad (4)$$

where  $N$  is the number of predefined categories,  $n_k$  is the number of categories that term  $t_k$  occurs in.

#### 3.2 The Utilities of Semantic Knowledge Bases for Email Classification

The naive VSM representation of emails is purely based on the occurrence of terms and totally neglects the semantic and syntactic properties of written text.



However, as explained in section one, the feature space for EC is far sparser than in conventional text classification due to the fact that emails are short in length, with noisy terms, and the training instances are usually too few. Therefore, the feature space based on naive VSM representation can't provide enough information for classification algorithms to learn accurate classifiers, and the generalization ability of these classifiers is also very limited due to the sparse feature space. To improve EC performance, additional resources like semantic knowledge base can be used. We use two semantic knowledge bases (one for Chinese terms and another for English terms) to integrating semantic features into the original VSM representation.

The two semantic knowledge bases used are WordNet and HowNet. WordNet is a large online English lexical database that contains various conceptual relations among words [10]. Its basic structure is a concept hierarchy that is composed of a set of nodes and relations among nodes. Each node is a 'synset' (i.e. synonymous set) corresponding to a particular concept and often consists of a number of terms or phrases. Nodes are correlated by various conceptual relations, including hypernymy, hyponymy, meronymy etc. Hyponymy means 'is a' or 'is a kind of' relation. Hypernymy is inverse of Hyponymy. For instance, lip stick is a hyponym of *makeup*.

HowNet is a public bilingual (Chinese-English) common-sense knowledge base describing relations among concepts as well as relations among the attributes of concepts [11]. The basic concept structure of HowNet is similar to Wordnet. The concepts in HowNet are also represented by a set of Chinese words, and there are also various relations among concepts including hypernymy/hyponymy, synonymy, attribute-host etc. The construction of HowNet is in a bottom-up fashion, and it takes advantage of the fact that all concepts can be expressed using a combination of one or more Chinese characters. HowNet covers about 65,000 concepts in Chinese. Though there are some differences between HowNet and WordNet, we use them in a similar way.

Generally, related concepts can be represented by a variety of words, and different authors of written text may employ diverse vocabulary. For instance, in our email dataset, the concept of 'transfer goods to customers' can be expressed by the word 'send', or 'ship', 'transport'. Therefore, integrating these features into the representation model will be helpful to enrich the sparse feature space of EC. Fortunately, free open semantic knowledge bases such as WordNet and HowNet greatly facilitate the process. With them the related semantic features such as the synonymies, hypernymy/hyponymy concept features can be imported to complement the original sparse feature space. This will result in a semantic-rich feature space that can provide more information for learning algorithms and in turn improve the classification performance. We proposed a FSE technique to carry out this process.

### 3.3 Enriched Feature Space for Email Classification

With the conceptual structure, the semantic knowledge bases can provide rich semantic features for information retrieval tasks. In previous work, they have

demonstrated their usefulness for term classification and text classification [12], which motivates us to employ them for enriching the highly sparse feature space of email classification. The main process of FSE is to expand the original feature space with the related semantic features from WordNet and HowNet. The resulted feature space of FSE is defined as follows:

$$\vec{V}_s = \vec{V}_o + \vec{V}_e \quad (5)$$

where  $\vec{V}_o$  is the original feature space defined in formula (1), and  $\vec{V}_e$  consists of the enriching features extracted from WordNet and HowNet.  $\vec{V}_o$  and  $\vec{V}_e$  are added to compose the resulted feature space  $\vec{V}_s$ . Based on the new populated feature space, the weight of each term in  $\vec{V}_s$  will be computed using the formula (3) and (4).

The core procedure of FSE is the selection of the enriching features, as there may be multiple synsets extracted from WordNet or HowNet for each original term, and only some of the synsets have the common concept with the original term. We propose the ‘information gain increment’ filtering metric to select the enriching features. The intuitive idea behind this scheme is to select only those extracted terms that will increase the global information for learning algorithms. The selection process consists of three steps as follows:

1. For each term in the emails, the synonym and direct hypernymy concepts (with all associated words) are extracted from WordNet (for English term) and HowNet (for Chinese term). Currently we just used hypernymy/hyponymy and synonymy relations, as they are the most common relation. During searching, only noun, adjective and verb words are looked up since they are considered semantic-richer.
2. For each extracted term  $t_k$ , the information gain of original feature space (denoted as  $IG(t_k)_o$ ) and the enriched feature space (denoted as  $IG(t_k)_e$ ) are calculated by the following formula [13]:

$$IG(t_k) = - \sum_{j=1}^{|C|} P(c_j) \log(P(c_j)) + P(t_k) \sum_{j=1}^{|C|} P(c_j|t_k) \log(P(c_j|t_k)) \\ + P(\bar{t}_k) \sum_{j=1}^{|C|} P(c_j|\bar{t}_k) \log(P(c_j|\bar{t}_k)) \quad (6)$$

where  $|C|$  is the total number of the categories,  $P(t_k)$  can be calculated from the fraction of emails in which the term  $t_k$  occurs and  $P(c_j)$  from the fraction of emails that belong to class  $c_j$ .  $P(c_j|t_k)$  is computed as the fraction of emails from class  $c_j$  that term  $t_k$  occurs and  $P(c_j|\bar{t}_k)$  as the fraction of emails from class  $c_j$  that term  $t_k$  does not occur.

3. Then the information gain increment of extracted term  $t_k$  is computed by:

$$\Delta IG(t_k) = IG(t_k)_e - IG(t_k)_o \quad (7)$$

if  $\Delta IG(t_k)$  is positive or higher than some predefined threshold, then the occurrence of extracted term  $t_k$  is used to populate the initial feature space.

The result of FSE is a semantic-richer feature space under which the TF\*IDF weighting algorithm is performed, and the naive VSM representation is converted into the enriched VSM representation for classification algorithms.

## 4 Experiments and Analysis

### 4.1 Classification Algorithms

We employed Naive Bayes (NB) [14], and SVM as the classification algorithms. The former is a well-known baseline learner, and SVM is one of the best-performed classifiers in text classification. Since a single SVM classifier can only solve two-class classification problems [15], multiple SVM classifiers must be built for multi-class classification. For  $N$ -class email classification we use the Decision Directed Acyclic Graph (DDAG) algorithm [16] to build an acyclic graph with  $N(N - 1)/2$  nodes, each of which is corresponding to a SVM classifiers. DDAG-SVM is very efficient and amenable to a VC-style bound of generalization error, as shown in [16].

### 4.2 Dataset

In this paper, we carried out experiments on a dataset from a large online cosmetic company. Every day the customer service center receives lots of emails from their customers who write emails in Chinese, English or both. The content of these emails varies from consultation for register problems to inquiry of product information, payment issues etc. To accelerate the service response, the company needs to automatically classify the emails and route them to corresponding personnel. The dataset contains 1586 emails collected from the company mail server and classified into six different categories. Table 1 shows the instance distribution of the dataset.

**Table 1.** Email Instance Distribution of The Dataset

Category	Email Number	Percentage
Register	76	4.792
Product	361	22.762
Order	553	34.868
Delivery	272	17.150
Payment	235	14.817
Feedback	89	5.611

### 4.3 Batch-Mode Experiment Results

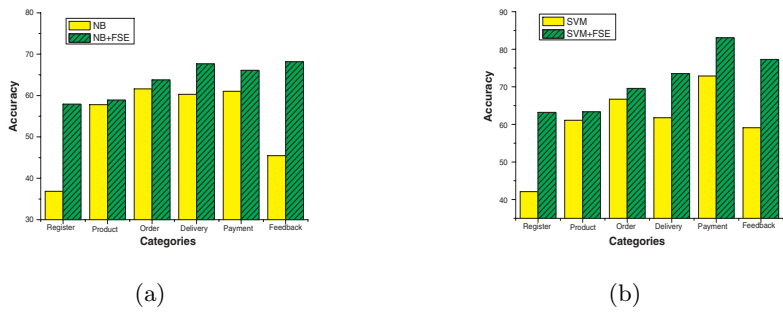
First, we ran a four-fold cross validation (CV) experiment on the dataset, the overall result is shown in table 2. Overall the classification accuracy of both NB

and SVM could be enhanced by our feature space expansion (FSE) method. The SVM classifier had an increment of 7.58 percent and was more than that of NB classifier which only obtained 5.31 percent increment. We consider that this is due to the fact that SVM has more global optimization and generalization ability so that it can integrate the new features more smoothly and effectively.

Figure 1 shows the detailed results of all categories. We can find that not all categories behave the same with FSE. Categories with few instances, such as ‘Register’ and ‘Feedback’, obtain more benefits from FSE. This is due to the fact that their original feature spaces are sparser and the FSE method can supply them more semantic features so that they can provide more discrimination information for learning algorithms. Categories with more emails seem to gain less improvement, such as the category ‘Product’ and ‘Order’. To explain this result, we analyzed the classification confusion matrixes, and found that after FSE more instances of the category ‘Product’ were misclassified into the category ‘Order’. Further inspection on their emails’ content revealed that they had quite a few terms in common and the FSE would expand the intersected term set too. In the end, the two categories might be made more confused. Since category ‘Order’ is more dominant, some similar instances of ‘Product’ would be classified as ‘Order’. This problem can be solved by using domain knowledge.

**Table 2.** The Overall Accuracy of the CV Experiment

Classifier	Without FSE(%)	With FSE(%)	Improvement(%)
Naive Bayes	58.33	63.64	5.31
SVM	63.38	70.96	7.58

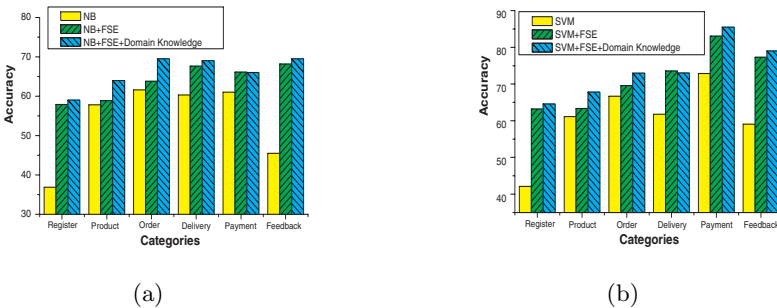


**Fig. 1.** Detailed Category Accuracy of the CV Experiment: figure (a) is the results of NB classifier, and figure (b) is the results of SVM classifier

Previous work has shown that domain knowledge is useful for enhancing the performance of text classification [8]. In our experiments, we found that some domain-specific terms with high frequency, such as proper nouns, could not be found in WordNet and HowNet due to the fact that these are general semantic

knowledge bases and don't include the words for special domains. To test if our FSE technique could benefit from the domain knowledge, we extended the WordNet and HowNet with domain knowledge and carried out some preliminary experiments. We expected that extending WordNet and HowNet with domain knowledge would further improve the EC performance.

Currently we just collected domain-specific terms manually from the email samples and some Web pages from cosmetic websites, and grouped them into some synsets based on their co-occurrence frequency in the sample documents. Some synsets like: {'Chanel', 'Christian Dior', 'Elizabeth Arden', ...} which is a synset for brand names, and {'Lumiere', 'Hydrabase', 'Hydrasoleil', ...} which is a synset for product types. These synsets make up of the domain extension for general semantic knowledge bases, and were used in the FSE process to populate the domain features in the original feature space. Then we ran a four-fold CV experiment on the dataset, and compared the results with that of Figure 1, the experiment results are illustrated in Figure 2.

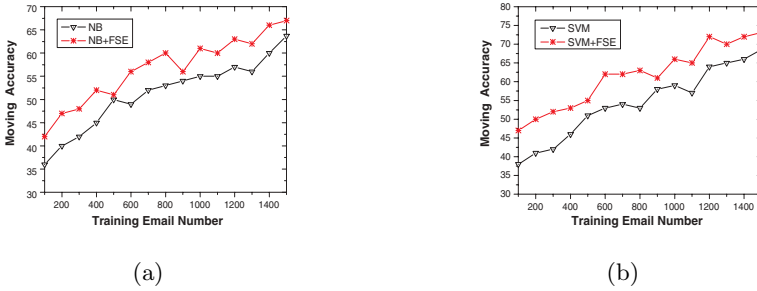


**Fig. 2.** Experiment Results of the Effect of Domain Knowledge

Overall, the domain knowledge is helpful for improving the classification accuracy. Especially, the category 'Product' and 'Order' benefited much more from it. As stated previously, the two categories are more confused, which presents a difficulty for FSE. However, with the support of domain knowledge the FSE technique could expand the domain features in the original feature space and increase their term weights, which in turn reinforced their discrimination ability. This implies that our FSE technique can be more effective for enhancing the classification accuracy with the extension of domain knowledge.

#### 4.4 Dynamic Experiment Results

Email domain is dynamic. In many cases training instances for email classification are not in large batch but in dynamic flow with possibly changing content, so the classification system should be able to learn incrementally. To address this problem, we carried out the dynamic experiments. At the beginning, according to the date of email the first 100 ones in the dataset were selected as



**Fig. 3.** Dynamic Experiment Results of NB Classifier and SVM Classifier

the initial training instances, then in every step the next 100 emails were taken out to be classified, the classification result of each step was recorded, and these emails were added to training set. This step repeated until the whole dataset had been tested. Figure 3 shows the experiment results on NB classifier and SVM classifier.

It is clear that our FSE method can enhance the classification accuracy of NB as well as SVM. Overall, the accuracy can be increased more during the early stage when the training instances are not enough and the FSE can complement it by integrating features from semantic knowledge bases. With the increasing of training instances, the improvement seems to slow down. As a whole, SVM classifier gains more benefits from FSE than the NB classifier that is similar to the CV experiment. Though the improvement of FSE varies during the dynamic process, FSE is helpful to EC persistently.

## 5 Conclusion

While previous work seldom addresses the problem of the sparse and noisy feature space for EC, this paper proposes a feature space enriching (FSE) technique to solve this problem. Two semantic knowledge bases, WordNet and HowNet, are employed to enable the enriching process. The enriched feature space provides semantic-rich feature space for our naive Bayes and SVM algorithms to learn improved classifiers. Experiments on bilingual enterprise email dataset show that the classification accuracy can be improved through this technique, while classes with relative few instances gain more remarkable improvement. We also find that FSE can further improve the classification accuracy with the support of domain knowledge. These experiments demonstrate the effectiveness of our FSE technique for improving EC performance.

There are some lines of work that can be extended in the future. One of them is to test FSE on other classification algorithms such as KNN, neural network etc. Secondly, the FSE technique can also be applied to other text classification tasks which have the similar problem of sparse feature space, such as short message classification, news filtering.

## References

1. Cohen, W.: Learning rules that classify e-mail. In: Proc. of the 1996 AAAI Spring Symposium on Machine Learning and Information Access. (1996)
2. Crawford, E., McCreath, E.: Iems: the intelligent email sorter. In: Proc. of the 19th International Conference on Machine Learning. (2002)
3. Provost, J.: Naive-bayes vs. rule-learning in classification of email. The University of Texas at Austin, Artificial Intelligence Lab, Technical Report (1999)
4. Rennie, J.D.: ifile: An application of machine learning to e-mail filtering. In: Proc. of KDD-2000 Text Mining Workshop. (2000)
5. Diao, Y., Lu, H., Wu, D.: A comparative study of classification-based personal e-mail filtering. In: Proc. of the PAKDD 2000. (2000)
6. Segal, R., Kephart, J.O.: Mailcat: An intelligent assistant for organizing e-mail. In: Proc. of the Third International Conference on Autonomous Agents. (1999)
7. Segal, R., Kephart, J.O.: Incremental learning in swiftfile. In: Proc. of the 17th International Conference on Machine Learning. (2000)
8. Brutlag, J., Meek, C.: Challenges of the email domain for text classification. In: Proc. of the 17th International Conference on Machine Learning. (2000)
9. Baeza-Yates, R., Ribeiro-Neto: Modern Information Retrieval. ACM Press Series/Addison Wesley,, New York (1999)
10. Miller, G.: Wordnet: a lexical database for english. *Communication of ACM* **38** (1995) 39–41
11. Dong, Z.: Bigger context and better understanding - expectation on future mt technology. In: Proc. of International Conference on Machine Translation and Computer Language Information Processing. (1999)
12. Gelbukh, A., Sidorov, G., Guzman-Arenas, A.: Use of a weighted topic hierarchy for document classification. In: *Lecture Notes in Artificial Intelligence*, No.1692: pp. 130-135, Springer-Verlag. (1999)
13. Yang, Y., Pedersen, J.O.: A comparative study on feature selection in text categorization. In: Proc. of the 14th Intl. Conference on Machine Learning. (1997)
14. Lewis, D.: Naive (bayes) at forty: The independence assumption in information retrieval. In: Proc. of the ECML-98. (1998)
15. Vapnik, V.N.: Statistical Learning Theory. John Wiley and Sons Inc. (1998)
16. Platt, J., Cristianini, N., Shawe, J.: Large margin dags for multiclass classification. In: *Advances in Neural Information Processing Systems*. (2000)

# PLD: A Distillation Algorithm for Misclassified Documents

Ding-Yi Chen and Xue Li

School of Information Technology and Electrical Engineering,  
University of Queensland, QLD 4072, Australia  
{dingyi,xueli}@itee.uq.edu.au

**Abstract.** We observed that in interactive text classification, user tends to point out only the misclassified documents, not the correct ones. It is unlikely that a user would be diligent enough to identify all the misclassified documents. In this case, a classifier is expected to deal with misclassified documents. Among them it is possible that only a small proportion has been identified. We propose the Prediction-Learning-Distillation (PLD) framework for distilling the misclassified documents. Whenever a user points out an error, the PLD learns from the mistake and identifies the same mistake from all other classified documents. The PLD then enforces this learning for future classifications. Our experiment results have demonstrated that the proposed algorithm can learn from user identified misclassified documents, then distills the rest successfully.

**Keywords:** Document Classification, SVM, Winnow Algorithm.

## 1 Introduction

Automatic text classification can contribute to the field such as Web directory constructing (e.g., for Yahoo and Google search engines), business document routing, and email filtering [1]. A classification decision is made based on *categorization status value* (CSV) [2], which represents the relevance between a document and a given category. The range of CSV is from 0 to 1, which indicates the strength of the relevance. A threshold  $\theta$  is usually given for decision making. Following formula shows how the classifier tells whether or not a document  $d$  is filed to category  $c$ :

$$h_c(d) = \begin{cases} 1, & \text{if } CSV_c(d) \geq \theta \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

If  $h_c(d) = 1$ , then  $d$  will be filed into  $c$ , otherwise  $d$  will not be filed into  $c$ .  $h_c(d)$  is also called *classification hypothesis* (CH), which is used by classifier to make classification decisions [3]. Conventional classification approaches such as Naive Bayes [4] and SVM [5] require training sets to work. This kind of approaches are also known as supervised classification. As the work [6] shows, 10 to 20 labeled documents are normally required for each category in order to provide sufficiently accurate results for supervised batch classification. However, an ordinary user



may not be able to explicitly articulate domain-knowledge, nor be capable of giving sufficient examples for building the training set as required.

We use a real-life example to illustrate a scenario of our research problem. In a company, a manager might ask a secretary to put a misfiled document into a correct folder, but if the secretary is doing the things right, probably no comments would be heard. Sometimes, the manager is too busy to point out the misfiled items. A qualified secretary should do three things to improve her/his performance:

- Reviewing all past classified documents to check out those misclassified documents that is similar to the one that has just been pointed out by the manager;
- Remembering the mistake that she/he has just made and never make the same/similar mistake in future;
- Trying to recognize the doubtful items and ask the manager for further instruction.

Compared with an interactive classification process, as the example shown, the manager stands for the user, while the secretary stands for a classifier program. The only feedback is a set of misclassified documents that are identified by user, which means a classifier could learn from only negative examples, that is, the misfiled documents. Since we can not expect that user is capable of revealing enough number of misclassified documents for conventional approaches, mining from the uncommented documents become essential. For mining latent information from uncommented documents, we define four important types of documents:

- *Identified Misclassified Documents (IMD)*: the misclassified documents identified by user.
- *Uncommented Documents (UD)*: the documents without user comments.
- *Uncommented Misclassified Documents (UMD)*: the misclassified documents which are not revealed by user but are identified by our learning program.
- *Uncommented Correctly classified Documents (UCD)*: the uncommented documents which are correctly classified.

These four terms are to be implemented as variable sets of documents. Initially they are all empty. After user feedback is received, the classifier will put documents together with user comments into IMD and proceed with the learning. The others are in UD. Our framework will then extract UMD and UCD from UD. In other words, if user points out some part of misclassified documents, our proposed framework will reveal other misclassified documents.

The idea of our algorithm is based on following observations:

1. Classification hypothesis (rules) should be changed after learning from the IMD.
2. Based on different ‘versions’ of classification hypotheses, the classification decisions on a certain uncommented document might be different. Should it happen, the documents might be in UMD, while the others are in UCD.

Obviously, observation 1 is true because IMD indicates that the original hypothesis does not fit the training instances. Therefore, CH should be updated to reflect the new training instances. Observation 2 is actually based on observation 1), the newer hypothesis tends to be more accurate for it has been trained with more training instances. Therefore, if the the newer classification decision on a given document is different from the older one, the document is likely to be misclassified by the older hypothesis.

We propose a framework to learn from user identified misclassified documents to distill the misclassified documents from uncommented ones. In the manager-secretary scenario, after manager pointing out some misclassified documents, the secretary then learns from the these misclassified documents and generate new knowledge (hypotheses). With these new knowledge, secretary is capable of discovering the misclassified documents which have not yet pointed out by the manager.

Our experiments shows that the misclassified documents can be successfully distilled even only a small proportion of errors has been identified by user; and our framework provide similar effectiveness with less number of training instances. The paper is organized as follows: Section 2 describes our proposed framework: PLD. Section 3 reviews the related work, especially the works which deal with large number of unlabeled data. Section 4 presents the experiment design and shows some promising results of the PLD framework. Section 5 concludes our contributions in the area of interactive document classifications.

## 2 The PLD Framework

The proposed framework, *Prediction-Learning-Distillation (PLD)*, is used to distinguish the uncommented correctly classified documents (UCD) and uncommented misclassified documents (UMD) from the uncommented documents (UD). The learning happens when the user identified misclassified documents are converted into the changes to the classification hypothesis as that shown in the DFD (Data Flow Diagram) in Fig. 1. The PLD framework has three components: predictor, learner, and distiller.

- **Predictor:** The component which classifies documents according to the current set of CH. Predictor uses the rules coded as  $h_c(d)$  (see equation 1) in CH to assign relevant category labels to each document.
- **Learner:** The component that updates CH from IMD.
- **Distiller:** The component that adds the documents into UCD and UMD from UD. The distiller generates new labels according to the updated classification function of  $h_c(d)$  for an uncommented document, then compare the new labels with the old ones, if the labels are identical, the document is identified as in UCD, otherwise in UMD.

Our research focus is not on how to classify documents but on how to learn from mistakes identified by the user in a process of interactive document classification. Therefore the PLD framework is designed to adopt different kinds of classification algorithms.

A classification algorithm performs two functions:

1.  $predict(d)$ : Classifier assigns category labels to an unlabeled document  $d$  according to CH.
2.  $train(D)$ : Classifier learns from set of labeled sample documents  $D$  and generates new CH.

We use the term, *core algorithm* to denote the classification algorithm which is adopted into PLD framework. The function  $predict(d)$  of core algorithm serves as predictor, while the function  $train(D)$  becomes learner. After the new CH is generated by the learner, the distiller filters out the unidentified misclassified documents by comparing the predicted labels (which are assigned by the predictor) to the new labels which are assigned by the function  $predict(d)$  according to the new CH. Technically speaking, a classification algorithm can be adopted as core algorithm of PLD framework by providing those functions. In order to test our framework, we use *online SVM* [7] and *balanced Winnow* [8] as examples of core algorithms.

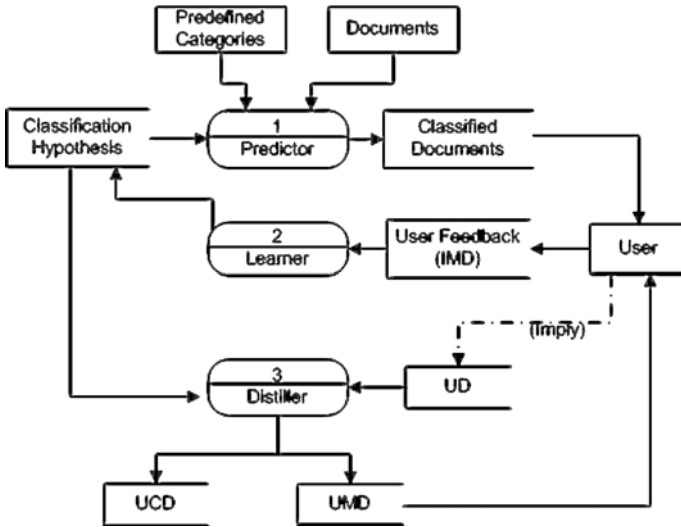


Fig. 1. The DFD of PLD framework.

The PLD framework works as follows (see Figure 1):

Whenever the incoming documents are received, the predictor labels these documents. At beginning, if supervised algorithms such as SVM and Winnow are adopted as core algorithms, they do not have any ‘knowledge’ (CH) to classify the documents, hence predictor does not put any labels on them. User then spends some time to read the documents and points out the misclassified ones to be sent to (IMD). The learner program then utilizes documents in IMD as

a training set and generates the new CH. The distiller program generates new labels according to the new CH, then distills the UMD and UCD according to the differences between new labels and the old predicted labels. The data flow diagram (DFD) of PLD is shown in Fig. 1 and the overall algorithm of PLD framework is given in Fig. 2.

**Input:** set of buffered documents  $D$ , category labels, IMD from user.  
**Output:** classification hypotheses,  
 set of UMD, and set of UCD  
**Algorithm:**

1. Initialize the classification hypotheses  $h_c(d)$ . (Depending on core algorithms)
2. The predictor assigns the ‘predicted’ category labels to each document according to  $h_c(d)$ .
3. The user provides the some correct decisions for misclassified documents (IMD).
4. The learner learns from IMD and update the  $h_c(d)$ .
5. The distiller assigns the new category labels to each document according to updated  $h_c(d)$ .
6. The distiller separates the UCD and UMD from UD by comparing the new labels and the labels assigned by predictor.

**Fig. 2.** The algorithm of PLD framework.

Please note that the PLD framework itself does not tell the details of learning. The one which actually takes care of learning and predicting is the core algorithm. Since different core algorithms have different ways to induct the hypotheses, the PLD provides a standard operating procedure to revised the mistakes which have not been identified yet.

### 3 Related Work

There are a few typical approaches that are able to work with a small amount of training instances and large amount of unlabeled data. Due to the relative rareness of labeled data, mining on the unlabeled documents becomes essential. In this case, we discuss the techniques of unlabeled document mining in this section. Also, since we adopt online SVM and balanced Winnow as core algorithms, the brief introductions of those classifiers will be given as well.

#### 3.1 Unlabeled Document Mining

How to handle the data without the user comments? Expectation Maximization (E-M) algorithm [9] uses the hypothesis from labeled documents to maximize

the expectation of the labels of unlabeled documents. In E-M algorithm, the initial classification hypotheses will be concluded from the labeled documents. On expectation step, classifier predicts the label for each unlabeled documents; on maximization step, classifier concludes the new classification hypotheses from both labeled and unlabeled documents. If the classification hypotheses become stable, then the iteration of E-M step will finish.

Mapping-Convergence (M-C) [10] algorithm can also be used to handle unlabeled data. M-C algorithm does not require user to provide negative training instances as conventional SVM does. Instead, in its mapping stage, the documents which do not have any positive features are treated as initial negative training instances (strong negative as referred in the work [10]); while convergence stage enlarges the negative training instances in order to provide more accurate result.

Similar to PLD, they use the different versions of CH in order to improve the effectiveness of classification tasks. However, the differences between conventional approaches and our approach are:

- Conventional approaches are usually bound to certain classifier, for example, M-C requires SVM to work; while PLD framework allows any classifiers.
- Conventional approaches work with unlabeled documents, while PLD framework deals with uncommented documents. The unlabeled documents are mere documents without labels, hence, no need to tell whether they are correctly classified or not; uncommented documents are documents with predicted labels, which means that the correctness of classification decisions become an important issue.
- Conventional approaches tend to use unlabeled documents to enlarge the training set; while PLD merely learns from IMD.

### 3.2 Classification Algorithms

The mistake-driven classifiers such as *positive Winnow* and *balanced Winnow* [8], and *Perceptron* [11] are often referred to, for their ability to learn from mistakes. Whenever a error has been identified, the mistake-driven classifier will modify the CH in order to correct the error. The reason we pick balanced Winnow is to neutralize the impact of the length of documents [12], for it is easier for longer documents tend to over the threshold in positive Winnow .

Due to the success of SVM in batch classification, several works [13–16] show their interests in applying SVM to online applications. The brief idea of online SVM is making a sliding window on the training data. After adding new training instances, the least significant training instances in the sliding window will be removed. Though they might not be as precise as their batch brethren, they are more suitable for large data set for they require less memory than batch ones.

## 4 Experiments and Results

The objective of our experiments is to examine the effectiveness of UMD detection. In order to examine the characteristic of our framework, we define a

independent variable, *diligence*, to denote the percentage of misclassified documents revealed by the user. *diligence* = 1 means all misclassified documents are pointed out by the user, while *diligence* = 0 means none of misclassified documents are identified. Since the effectiveness of UMD detection may be affected by how many IMD are revealed, it is the reason to introduce *diligence* into our experiments. Please note we skip the experiment with *diligence* = 0, because PLD can not get any training instances if none of the misclassified documents are revealed.

The effectiveness of classification decision and UMD detection can be measured by  $F_1$  function [17], where

$$F_1 = \frac{2(\text{Precision}) \bullet (\text{Recall})}{(\text{Precision}) + (\text{Recall})}, \quad (2)$$

Precision and recall of UMD decision are defined as follow:

$$\text{Precision} = \frac{\text{number of UMD which are correctly identified by PLD}}{\text{number of UMD which are identified by PLD}}, \quad (3)$$

$$\text{Recall} = \frac{\text{number of UMD which are correctly identified by PLD}}{\text{actual number of UMD}}. \quad (4)$$

Please note that if *diligence* = 1, precision will be zero, for there is no UMD. and recall can not be defined, for denominator is zero.

#### 4.1 Experiment Design

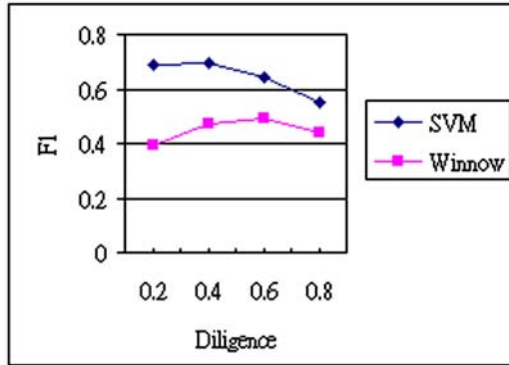
The modified Apte (“ModApte”) split on Reuters-21578 [18] corpus is used to split corpus into a training set (with 9,603 documents) and a test set (with 3,299 documents). The Porter Algorithm [19] is applied for stemming the postfix of words in order to aggregate the relative words. When processing the training set, PLD assigns category labels to the documents in buffer, if the labels differ from the *true labels* (the labels shown in corpus), then these documents are misclassified. The examiner (the program which examines classifier) separates the misclassified documents into IMD and UMD according to the *diligence* value. IMD is learned as described in algorithm; while the UMD and correctly classified documents are merged together as unclassified documents. The distiller then splits the unclassified documents into UCD and UMD. The precision and recall of UMD detection can be calculated after distillation.

When processing the test set, the learner ceases learning, only predictor assigns labels to test documents. The precision and recall of classification decision making can be scored after test set is finished.

#### 4.2 Results

In our experiments, online SVM and balanced Winnow are both used as core algorithms of PLD framework separately in order to test the effectiveness of PLD, and whether the PLD is algorithm dependent or not.

Figure 3 is particularly used to show the differences of effectiveness between online SVM and balanced Winnow algorithms in the distillation of set UMD. Online SVM shows a better result than the balanced Winnow algorithm, especially when diligence is low. Another observation we have on Fig. 3 is that the PLD framework is algorithm dependent, which means that the core algorithms selection does affect the distillation effectiveness of PLD.

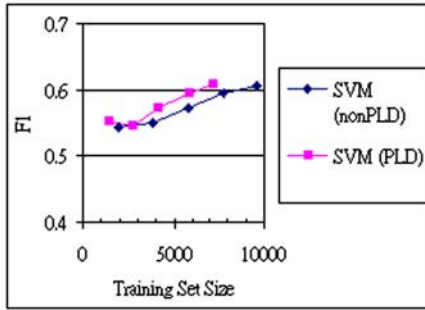


**Fig. 3.** UMD distillation comparison between online SVM and balanced Winnow.

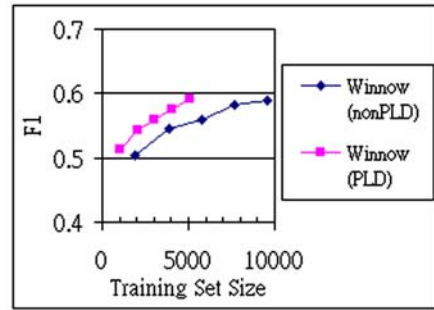
Figure 4 and 5 shows the effectiveness comparison between PLD and non-PLD. Dots shown in figure 4 and 5 stand for effectiveness on certain percentage of training instances revealed to classifier. For PLD, the dots stand for diligence=20%, 40%, 60%, 80% and 100%, from left to right; and for non-PLD, they stands for 20%, 40%, 60%, 80% and 100% of training instances which have been seen by the classifier. The reason that the PLD curves are shorter is that PLD only require misclassified documents instead of whole training set. For example, the whole training set contains 9603 documents, while Winnow only needs 5094 documents to reach the 100% diligence. As figure 4 and 5 shown, PLD requires less training instances, and the effectiveness is not sacrificed.

## 5 Conclusion

The ultimate objective of our research is to find a better way to maximize the effectiveness of the classification tasks, while minimize the required user involvement. As our experiments showed, our approach successfully reveals the un-commented misclassified documents even on low diligence, and same effectiveness can be achieved with fewer training instances. The basic idea of the PLD is: if any mistake is pointed out, then PLD utilizes the knowledge inducted by core algorithm to filter out the same/similar mistakes that have not yet been identified. As described in the manager-secretary scenario, after misclassified documents



**Fig. 4.** Effectiveness of SVM without/with PLD.



**Fig. 5.** Effectiveness of Winnow without/with PLD.

being pointed out, the secretary then obtains the further knowledge of classification and use the knowledge to discover the rest of the same/similar mistakes. Thus, this technique is implemented to improve the effectiveness of document classification. Our contributions are:

1. Proposed a novel approach to identify the uncommented misclassified documents by learning from the identified misclassified documents.
2. Providing a way to reduce the degree of human involvement, without losing the effectiveness.

After the distillation, the discovered UMD can be sent back to the user in order to retrieve further instructions and refine the overall effectiveness of the classification task.

We are investigating the further improvement of UMD distillation, as well as the utilization of distilled UMD in other classification algorithms.

## References

1. Chakrabarti, S., Dom, B., Indyk, P.: Enhanced hypertext categorization using hyperlinks. In: Proceedings of the 1998 ACM SIGMOD international conference on Management of data, Seattle, Washington, United States, ACM Press (1998) 307–318
2. Sebastiani, F.: Machine learning in automated text categorization. *ACM Computing Surveys (CSUR)* **34** (2002) 1–47
3. Blum, A., Mitchell, T.: Combining labeled and unlabeled data with co-training. In: Proceedings of the eleventh annual conference on Computational learning theory, Madison, Wisconsin, United States, ACM Press (1998)
4. Lewis, D.D.: Naive (bayes) at forty: The independence assumption in information retrieval. In: Proceedings of ECML-98, 10th European Conference on Machine Learning, Chemnitz, DE, Springer Verlag, Heidelberg, DE (1998) 4–15
5. Joachims, T.: Text categorization with support vector machines: learning with many relevant features. In: Rouveirol, C.N., Celine, eds.: Proceedings of ECML-98, 10th European Conference on Machine Learning, Chemnitz, DE, Springer Verlag, Heidelberg, DE (1998) 137–142



6. Tresch, M., Luniewski, A.: An extensible classifier for semi-structured documents. In: *Proceedings of the fourth international conference on Information and knowledge management*, Baltimore, Maryland, United States, ACM Press (1995) 226–233
7. Tax, D.M., Laskov, P.: Online svm learning: From classification to data description and back. In: *IEEE International Workshop on Neural Networks for Signal Processing (NNSP)*, Toulouse France (2003) 499–508
8. Littlestone, N.: Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning* **2** (1988) 285–318
9. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society* **39** (1977) 1–38
10. Yu, H., Han, J., Chang, K.C.C.: Pebl: positive example based learning for web page classification using svm. In: *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, Edmonton, Alberta, Canada, ACM Press (2002) 239–248
11. Schutze, H., Hull, D.A., Pedersen, J.O.: A comparison of classifiers and document representations for the routing problem. In: *Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*, Seattle, Washington, United States, ACM Press (1995) 229–237
12. Yamazaki, T., Dagan, I.: Mistake-driven learning with thesaurus for text categorization. In: *Proceedings of NLPRS-97, the Natural Language Processing Pacific Rim Symposium*, Phuket, TH (1997) 369–374
13. Cauwenberghs, G., Poggio, T.: Incremental and decremental support vector machine learning. In: *Advances in Neural Information Processing Systems*. (2000) 409–415
14. Kivinen, J., Smola, A.J., Williamson, R.C.: Online learning with kernels. In: *Advances in Neural Information Processing Systems*. (2001) 785–792
15. Ralaivola, L., d'Alché Buc, F.: Incremental support vector machine learning: A local approach. *Lecture Notes in Computer Science* **2130** (2001) 322–330
16. Syed, N.A., Liu, H., Sung, K.K.: Incremental learning with support vector machines. In: *Proceedings of the Workshop on Support Vector Machines at the International Joint Conference on Artificial Intelligence (IJCAI-99)*, Stockholm, Sweden (1999)
17. Van Rijsbergen, C.J.: Evaluation. In: *Dept. of Computer Science, University of Glasgow. Department of Computer Science, University of Glasgow* (1979)
18. Lewis, D.D.: Reuters corpus (21578) (2000)
19. Porter, M.F.: An algorithm for suffix stripping. *Program* **14** (1980) 130–137

# Sequential Classifiers Combination for Text Categorization: An Experimental Study\*

Zheng Zhang, Shuigeng Zhou, and Aoying Zhou

Dept. of Computer Science and Engineering, Fudan University, Shanghai, 200433, China  
{zhzhang1981, sgzhou, ayzhou}@fudan.edu.cn

**Abstract.** In this paper, we introduce *Sequential Classifiers Combination* (SCC) into text categorization to improve both the classification effectiveness and classification efficiency of the combined individual classifiers. We apply two classifiers sequentially for experimental study, where the first classifier (called *filtering classifier*) is used to generate candidate categories for the test document and the second classifier (called *deciding classifier*) is used to select a final category for the test document from the candidate categories. Experimental results indicate that when combining boosting and  $k$ NN methods, the combined classifier outperforms the best one of the two individual classifiers, and in the case of combining Rocchio and  $k$ NN methods, the combined classifier performs equally well as  $k$ NN while its efficiency is much better than  $k$ NN and is close to that of Rocchio.

## 1 Introduction

With the rapid growth of the amount of documents in digital form, how to manage these massive documents repositories and retrieve what we want from them accurately and efficiently is a severe challenge to both database and IR communities. Automated text categorization (TC, also known as text classification), the task of assigning predefined category labels to new documents based on the rule suggested by a training set of documents, is one method that can help to solve this problem. So far, a lot of learning algorithms have been proposed for text categorization, including naïve Bayes classifier [1], decision tree [2],  $k$ NN classifier [3], online classifier [4], Rocchio classifier [5], support vector machine [6] and boosting method [7] etc.

In addition to developing different TC methods, some other researches try to combine multiple classifiers for achieving better performance than any individual classifier can obtain. These methods are termed multiple classifiers combination. Current combination methods include majority voting (MV) [9], weighted linear combination (WLC) [9], dynamic classifier selection (DCS) [10] and adaptive classifier combination (ACC) [11] etc. In these combination methods, different individual classifiers make judgments separately on a test document according to their respective classifying algorithms, and the final (usually better) judgment is made based on these judg-

---

\* This work was supported by National Natural Science Foundation of China under grant No. 60373019.

ments according to a pre-specified combination rule (e.g. MV, WLC, DCS and ACC). Because these different individual classifiers perform classification independently, we can call these methods *parallel combination* methods.

Different from the parallel combination methods, *Sequential Classifiers Combination* (SCC) [16] employs multiple classifiers sequentially in the same categorization task where the latter classifiers utilize the results of the former ones. However, prior to this paper, Sequential Classifiers Combination has not been used in text categorization field. We conduct an experimental study to validate the performance of this method in text categorization. In our experimental study,  $k$ NN classifier is combined with Rocchio classifier and Boosting classifier respectively. The experimental results indicate that when we combine Boosting and  $k$ NN, the SCC classifier outperforms the best individual classifier; while we combine Rocchio and  $k$ NN, the SCC classifier performs equally well as  $k$ NN, but its efficiency is much better than  $k$ NN and is close to that of Rocchio.

The rest parts of this paper are organized as follows. Section 2 is the related work about text classifiers combination methods; Section 3 introduces sequential classifiers combination (SCC) method; Section 4 presents an experimental study of SCC method; Section 5 summarizes our experiment results of SCC method; Section 6 concludes the paper and highlights some future work directions.

## 2 Related Work

Text categorization has been extensively studied in the last decade, and a lot of methods were proposed. Typical text categorization methods include naïve Bayes classifier [1], decision tree [2],  $k$ NN classifier [3], neural network [4], Rocchio classifier [5], support vector machine (SVM) [6] and boosting method [7] etc. For individual classification approach, we refer the readers to [8, 12]. To further improve categorization performance, approaches to combine different classifiers in text categorization were also investigated, such as majority voting (MV) [9], weighted linear combination (WLC) [9], dynamic classifier selection (DCS) [10] and adaptive classifier combination (ACC) [10]. In majority voting (MV), classification judgments for a test document obtained by the  $k$  classifiers are pooled together, and the classification decision that reaches the majority (*i.e.*, at least  $(k+1)/2$  votes) is taken ( $k$  obviously needs to be an odd number). In weighted linear combination (WLC), different classifiers first calculate scores of each category for the test document individually and then get a weighted sum of these scores. The weight  $w_j$  means the expected relative effectiveness of classifier  $\Phi_j$ , and the sum of all weights should be equal to 1. These weights are assigned by the user, and typically are optimized on a validation set. The category with the largest summed score is taken as the final result. As for dynamic classifier selection (DCS), among classifier committee the individual classifier that performs best on the  $l$  training documents most similar to the test document  $d$  is selected, and its judgment for  $d$  is adopted eventually by the committee. In adaptive classifier combination (ACC), the judgments of all the classifiers in the committee are summed together, but their individual contribution is weighted by the effectiveness

that they have shown on the  $l$  training documents most similar to the test document  $d$ . So ACC can be seen as a kind of hybrid between DCS and WLC.

3 Sequential Classifiers Combination

All the existing text combination methods are essentially *parallel* methods, that is, different classifiers make their judgments on a test document using their individual classifying algorithms *separately* and the final decision is then made based on these judgments according to a certain rule (e.g. MV, WLC, DCS and ACC), see Fig. 1(a) for illustration. In contrast to parallel combination, here we introduce Sequential Classifiers Combination (SCC) into text categorization field, that is, two or more classifiers classify the document sequentially where the latter classifier utilizes the categorization results of the former ones as input. See Fig. 1(b) for illustration.

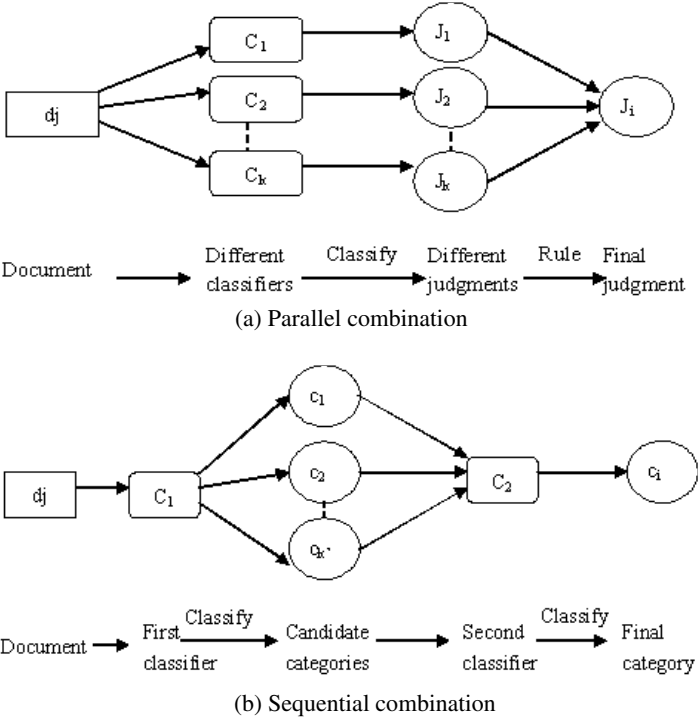


Fig. 1. Illustration of parallel combination and sequential combination

In Fig. 1(b), the process of classifying a document with our sequential classifiers combination method is illustrated. Here two different classifiers are combined, namely,  $C_1$  and  $C_2$ .  $C_1$  is used to generate a certain number of *candidate categories* for a test document  $d_j$ , say  $c_1, c_2, \dots, c_k$ , and  $C_2$  is then used to select the final category (say  $c_i, 1 \leq i \leq k$ ) from the candidate categories generated by  $C_1$ . Intuitively, we

call the first classifier *filtering classifier*, which is responsible for filtering out a relatively large number of categories where the test document does not belong to with high probability; and the second classifier is termed *deciding classifier*, which makes the final categorization decision on the test document within a relatively small number of candidate categories. The number of candidate categories generated by the filtering classifier (denoted as  $k'$ ) is a system parameter in SCC, which can be selected by users or tuned to yield the optimal performance by experiments.

To achieve good categorization performance with regard to both effectiveness and efficiency, we should be careful about the configuration of SCC. That is, how to select *filtering classifiers*, *deciding classifier*, and the number of candidate categories  $k'$  generated by the filtering classifiers. Intuitively, for the *deciding classifier*, we should pay more attention on its precision than on its efficiency for it makes the final categorization decision. However, for filtering classifiers, we don't require them to have very high precision but their efficiency must be emphasized because their task is to select some candidate categories (instead of the final result) from the whole category space. More formally, let say the combination is  $SCC=\{C_1, C_2, \dots, C_k\}$ , here  $C_1, C_2, \dots, C_{k-1}$  are *filtering classifiers*,  $C_k$  is the *deciding classifier*. Denote  $p(C_i)$  and  $e(C_i)$  the precision and efficiency of classifier  $C_i$  respectively, then a reasonable configuration of SCC should like this:  $p(C_1) \leq p(C_2) \leq \dots \leq p(C_k)$ , and  $e(C_1) \geq e(C_2) \geq \dots \geq e(C_k)$ . As for the setting of  $k'$  value, a general rule is like this: if the filtering classifier has high precision, then we can select a small value for  $k'$ , otherwise, we should select a larger one.

## 4 Sequential Classifier Combination for Text Categorization

To demonstrate the performance of SCC method for TC task, we conduct an experimental study in this section. Our goal is to improve  $kNN$  classifier's performance by combining it with other two classifiers: Rocchio classifier and Boosting classifier. We establish two SCC classifiers:  $SCC1=\{\text{Rocchio}, kNN\}$ ,  $SCC2=\{\text{Boosting}, kNN\}$ . That is,  $kNN$  is used as deciding classifier, Rocchio classifier and Boosting classifier play the role of filtering classifier. Experimental results show that SCC classifiers outperform  $kNN$  classifier in both classification precision and classification efficiency.

$kNN$  is one of the most effective methods in text classification, but it suffers from the problem of efficiency which is induced by two reasons: first,  $kNN$  is an example-based or lazy method, that is, it defers the decision on how to generalize beyond the training data until each new query instance is encountered [8]; second, in order to find the  $k$  documents which are most similar to the test document  $d$ ,  $kNN$  must compare  $d$  with all the documents in the training set based on the cosine formula using the TF-IDF weighing scheme. The time complexity of  $kNN$  algorithm is  $O(|D|N_t)$ , where  $|D|$  is the number of the training documents and  $N_t$  is the number of the test documents. In contrast to  $kNN$ , the Rocchio classifier, whose classification algorithm is quite similar to  $kNN$ , is much more efficient than  $kNN$ . This is because Rocchio compares the test document  $d$  with the "profile" of each category rather than with all

the documents in the training set. The time complexity of Rocchio is  $O(|C|N_t)$ , where  $|C|$  is the number of categories. Typically,  $|C|$  is quite smaller than  $|D|$ . Although Rocchio is a very efficient method, its classification precision is rather poor compared to  $k$ NN. So combining  $k$ NN and Rocchio can merge the advantages of  $k$ NN's precision and Rocchio's efficiency.

The motivation of combining  $k$ NN and Boosting is to demonstrate the general nature of the SCC method since the learning algorithm of Boosting is quite different from that of Rocchio and  $k$ NN and its performance is well accepted. Following we give brief introduction to  $k$ NN, Rocchio and Boosting classification methods respectively.

#### 4.1 $k$ NN Classifier

$k$ NN classifier (also known as  $k$ -nearest neighbor classifier), which is based on the term vector space model, is an effective method in text categorization. Given a test document  $d$ ,  $k$ NN's classification process consists of two steps. In the first step, the classifier finds  $k$  documents from the training set which are most similar to  $d$ . Usually similarity between two documents is calculated according to the cosine formula based on the TF-IDF weighing scheme [13]. In the second step, the classifier assigns a score to each category and ranks the candidate categories according to the score of each category. In a single-label case, the top category in the ranking list is the category  $k$ NN assigns to the test document. Formally, for a test document  $d$ , we calculate the score of category  $c_i$  as follows:

$$\text{score}(d, c_i) = \sum_{d_j \in kNN(d) \cap d_j \in c_i} \text{sim}(d, d_j) - b_i \quad (1)$$

$kNN(d)$  in the formula is the set of  $k$  documents nearest to  $d$ ,  $b_i$  is a threshold for each category, which can be tuned on the training set.

#### 4.2 Rocchio Classifier

Rocchio classifier is a TC method whose root lies exclusively in the IR tradition, and is a typical example of profile-based classifiers that extract an explicit profile of each category from the training set. The Rocchio method computes a classifier  $(w_{1i}, w_{2i}, \dots, w_{ri})$  for category  $c_i$  by the following formula[8]:

$$w_{ki} = \left( \frac{\beta}{|\{d_j \mid ca_{ij} = 1\}|} \cdot \sum_{\{d_j \mid ca_{ij} = 1\}} w_{kj} \right) - \left( \frac{\gamma}{|\{d_j \mid ca_{ij} = 0\}|} \cdot \sum_{\{d_j \mid ca_{ij} = 0\}} w_{kj} \right) \quad (2)$$

In this formula,  $ca_{ij}=1$  means that the document  $d_j$  belongs to the category  $c_i$ , while  $ca_{ij}=0$  indicates that  $d_j$  doesn't belong to the category  $c_i$ . The documents that belong to the category  $c_i$  are called "positive" examples of  $c_i$ , and the documents doesn't belong to  $c_i$  are called "negative" examples of  $c_i$ , and are control parameters that allow setting the relative importance of positive and negative examples. When setting  $\beta=1$  and  $\gamma=0$ , the profile of each category is exactly the centroid of the positive train-

ing examples. To assign a category to a document  $d$ , it comes down to calculate the similarity of  $d$  with the profile of each category and select the most similar category.

This method is quite easy to implement, and the resulting classifier tends to be quite efficient compared to  $k$ NN classifier. However, in terms of classification effectiveness, it is not a good TC method.

### 4.3 Boosting

Boosting is originally a machine learning technique and later introduced into text classification by Schapire [7]. The main idea of boosting is to combine many simple and moderately inaccurate categorization rules into a single, highly accurate categorization rule. We must point out that the concept of combination used by boosting is quite different from the combination we proposed in this paper: the combination in boosting is in the step of constructing the classifier while the combination we discuss in this paper is in the step of classifying documents with classifiers.

There are  $k$  different simple and moderately inaccurate classifiers in boosting method. These  $k$  classifiers are trained sequentially, *i.e.*, one after another. The training of each classifier will take into account how previous classifiers perform on the training set, and concentrate on getting right categories on those documents where previous classifiers perform worst. After all the  $k$  classifiers have been constructed, a weighted linear combination rule is applied to yield the final classifier.

## 5 Experimental Results

The Chinese text collection used in our experiment consists of news reports from the *People's Daily*. The whole data set has 2850 documents, which are categorized into 20 groups manually. We call this data set G1. In order to experiment on different data sets, we divide G1 into G2 and G3. Table1 shows the name of each category and the corresponding number of the documents in each category. The documents that belong to the 10 categories in the first and third column of Table1 form data set G2 and G3, respectively, while the whole documents in Table1 form data set G1.

**Table 1.** Experimental data set

Category	No. of documents	Category	No. of documents
Politics (G2)	617	Mine (G3)	67
Sports (G2)	350	Military (G3)	150
Economy (G2)	226	Computer (G3)	109
Agriculture (G2)	86	Electronic (G3)	55
Environment (G2)	102	Communication (G3)	52
Space (G2)	119	Energy (G3)	65
Art (G2)	150	Philosophy (G3)	89
Education (G2)	120	History (G3)	103
Medic (G2)	104	Law (G3)	103
Transport (G2)	116	Literature (G3)	67

In our experiment, for each category we choose 70% of the documents in the data set as the training set and the left 30% as the test set. Each test in our experiment is repeated for 5 times and each time we select the training documents and test documents randomly. We get the final result by averaging results of each trial.

We preprocess the Chinese documents by removing the low-frequency words, that is, the words appear less than 2 times in a document. We don't remove the stop-words, which will be filtered by our feature selection algorithm.

There is a challenge on how to get Chinese words or features from Chinese documents. In English documents, the words or features are separated naturally by blanks and thus are easily obtained, while in Chinese documents the words are Chinese characters string without separation between them and thus are difficult to extract. Generally, there are two different methods to obtain words from Chinese documents. One method is to use segmentation procedure with dictionary support [13]. This method is very complex and its accuracy is not proportionate to its complexity. The other method is to use N-gram [14] as the features of Chinese documents. Although the semantics of N-gram is less accurate than the real word, this method is quite efficient. In our experiment, we use the second method to get features from Chinese documents.

We use the global dimensionality reduction (DR) technique and information gain (IG) [8] method to reduce the high dimensionality of the term space. If not specially mentioned, the number of features in our experiment is 500. We use macro-average F1 and micro-average precision [8] as the evaluation measures of the text classifiers.

Table 2 shows a comparison of the performances of 3 different classifiers on our data set G1, G2 and G3 respectively. All the parameters for different classifiers are tuned to yield the best performance. For  $k$ NN, we set  $k=10$ (neighborhood size) in data set G2, G3, and  $k=20$  in G1. For boosting, we set training round to 400 in all of these three data sets. The experimental results indicate that  $k$ NN performs best on our Chinese dataset, and boosting performs worst.

**Table 2.** Performance of three different classifiers

	Macro-average F1			Micro-average precision		
	G1	G2	G3	G1	G2	G3
Rocchio	0.753	0.856	0.780	0.779	0.889	0.751
$k$ NN	<b>0.772</b>	<b>0.898</b>	<b>0.794</b>	<b>0.815</b>	<b>0.931</b>	<b>0.798</b>
Boosting	0.588	0.787	0.706	0.634	0.822	0.705

Results of sequential combination of Rocchio and  $k$ NN are shown in Fig. 2, 3 and 4. Fig. 2 and 3 show effectiveness of individual classifiers and combined classifier, evaluated by Macro-average F1 and Micro-average precision respectively. Fig. 4 shows efficiency of each classifier evaluated by time cost for classification. This test is over G1, but the trends performed on G2 and G3 are very similar. In G1, there are 20 categories in total. The X-axis of each figure represents the number of candidate categories in SCC (denoted by  $k'$ ). In our experiment, we choose Rocchio as the filtering classifier and  $k$ NN as the deciding classifier.



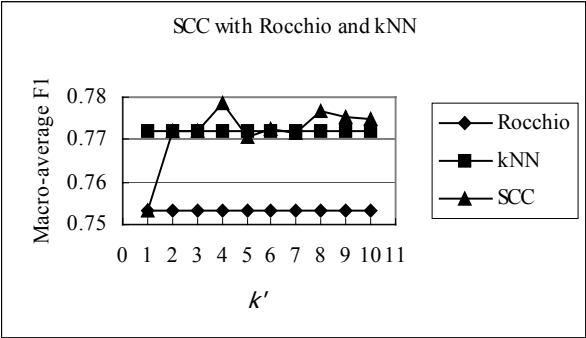


Fig. 2. Macro-average F1 of kNN, Rocchio and SCC with different  $k'$

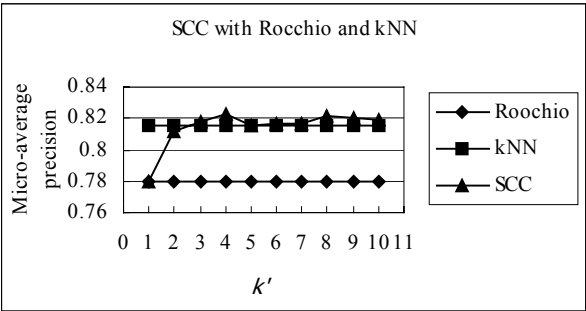


Fig. 3. Micro-average precision of kNN, Rocchio and SCC with different  $k'$

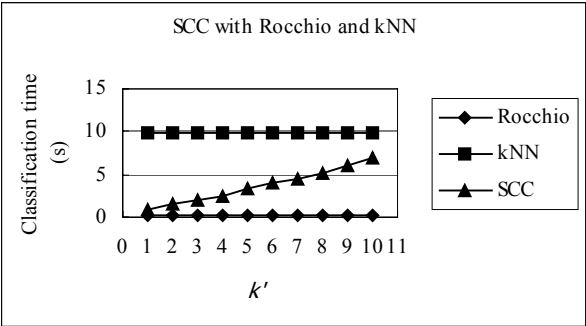


Fig. 4. Classification time of kNN, Rocchio and SCC with different  $k'$

Fig. 2 and 3 indicate that when  $k'=1$ , the effectiveness of SCC is equal to Rocchio; but when  $k'=2$ , the effectiveness of SCC improves radically and is close to kNN; when  $k'=4$ , SCC outperforms kNN. The effectiveness of SCC when  $k'=11\sim 20$  are not shown in Figure 2 and 3, but we can figure it out that as the value  $k'$  gets close to 20, the effectiveness of SCC is tending equal to kNN. Fig. 4 shows that the classification time for SCC is nearly a linear function of  $k'$ . From figure 2, 3 and 4, we can see that when  $k'=4$ , SCC outperforms the best individual classifier (*i.e.*, kNN), but its classifi-

cation time is about one fourth of that of  $k$ NN. Fig. 5 shows experimental results of combining  $k$ NN and Boosting by SCC where boosting is the filtering classifier and  $k$ NN is the deciding classifier. From Fig. 5, we can see that when  $k'=4$ , SCC yields the best effectiveness and outperforms the best individual classifier (*i.e.*,  $k$ NN) by more than 2 percent. This test is done on data set G3, and the number of total categories in G3 is 10. We also investigate performance of SCC and individual classifiers with different features sizes. Experimental results indicate that SCC performs best with medium-sized features. Fig. 6 shows the micro-average precision with different numbers of features on data set G1. The parameter  $k'$  in SCC is tuned to yield the best effectiveness for different numbers of features.

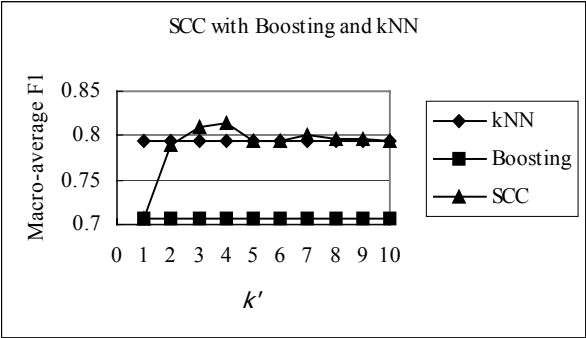


Fig. 5. Macro-average F1 of  $k$ NN, Boosting and SCC with different  $k'$  values

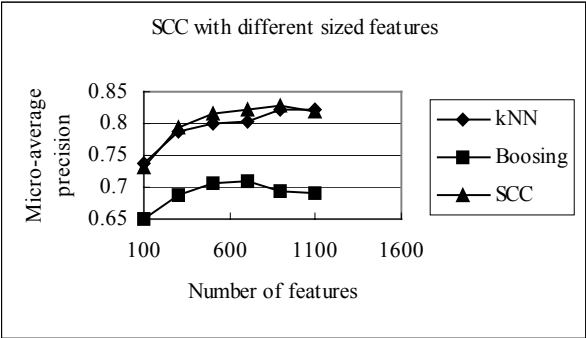


Fig. 6. Micro-average precision of  $k$ NN, Boosting and SCC for different feature sizes

## 6 Conclusion Remarks

In this paper, we introduce *Sequential Classifiers Combination* (SCC) into text categorization filed and conduct an experimental study to test its performance. The experiment results indicate that the SCC method can improve both the efficiency and effectiveness of individual text classifiers. As for future work, first, we will test the SCC method on more text collections, including standard benchmark collections such

as Reuters collections; second, we would like to test more SCC classifiers by combining different classifiers; and third, we plan to do a comprehensive comparison between SCC method and parallel combination method.

## References

- [1] Lewis, D. D. 1992. An evaluation of phrasal and clustered representations on a text categorization task. In *Proceedings of SIGIR-92, 15th ACM International Conference on Research and Development in Information Retrieval* pp. 37-50.
- [2] Lewis, D. D. and Ringuette, M. 1994. A comparison of two learning algorithms for text categorization. In *Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval* (Las Vegas, US, 1994), pp. 81-93.
- [3] Yang, Y. 1994. Expert network: effective and efficient learning from human decisions in text categorization and retrieval. In *Proceedings of SIGIR-94, 17th ACM International Conference on Research and Development in Information Retrieval* (Dublin, IE, 1994), pp. 13-22.
- [4] Wiener, E., Pedersen, J. O., and Weigend, A. S. 1995. A neural network approach to topic spotting. In *Proceedings of SDAIR-95, 4th Annual Symposium on Document Analysis and Information Retrieval* (Las Vegas, US, 1995), pp. 317-332.
- [5] Hull, D. A. 1994. Improving text retrieval for the routing problem using latent semantic indexing. In *Proceedings of SIGIR-94, 17th ACM International Conference on Research and Development in Information Retrieval* (Dublin, IE, 1994), pp. 282-289.
- [6] Joachims, T. 1998. Text categorization with support vector machines: learning with many relevant features. In *Proceedings of ECML-98, 10th European Conference on Machine Learning* (Chemnitz, DE, 1998), pp. 137-142.
- [7] Schapire, R. E., Singer, Y., and Singhal, A. 1998. Boosting and Rocchio applied to text filtering. In *Proceedings of SIGIR-98, 21st ACM International Conference on Research and Development in Information Retrieval* (Melbourne, AU, 1998), pp. 215-223.
- [8] Yang, Y. and Liu, X. 1999. A re-examination of text categorization methods. In *Proceedings of SIGIR-99, 22nd ACM International Conference on Research and Development in Information Retrieval* (Berkeley, US, 1999), pp. 42-49.
- [9] Larkey, S. and Croft, W. 1996. Combining classifiers in text classification. In *Proc. SIGIR*, pp. 81-93.
- [10] Woods, K., Kegelmeyer, W. P., Jr., and Bowyer Jr, K. 1997. Combination of multiple classifiers using local accuracy estimates. *IEEE Trans. PAMI*. 19(4): 405-410.
- [11] Li, Y. H. and Jain, A. K. 1998. Classification of text documents. *Computer. J.* 41(8): 537-546.
- [12] Sebastiani, F. 2002. Machine learning in automated text categorization. *ACM Computing Surveys*. 34(1): 1-47.
- [13] Salton, G. and McGill, M. 1983. *Introduction to Modern Information Retrieval*. McGraw-Hill, New York.
- [14] Zhou, S., Guan, J. and Hu, Y. 2002. Chinese documents classification based on N-grams. In *Proceedings of Third International Conference on Intelligent Text Processing and Computational Linguistics (CICLing-2002)*, LNCS. 2276, 405-414.
- [15] Larkey, L. S. 1998. Automatic essay grading using text categorization techniques. In *Proceedings of SIGIR-98, 21st ACM International Conference on Research and Development in Information Retrieval* (Melbourne, Australia, 1998), 90-95.
- [16] Rahman, A. F. R. and Fairhurst, M. C. 1999. Serial Combination of Multiple Experts: A Unified Evaluation. *Pattern Anal. Appl.* 2(4): 292-311.

# Combining Active Learning and Boosting for Naïve Bayes Text Classifiers<sup>\*</sup>

Han-joon Kim<sup>1</sup> and Je-uk Kim<sup>2</sup>

<sup>1</sup> Department of Electrical and Computer Engineering, University of Seoul, Korea  
khj@uos.ac.kr

<sup>2</sup> Daewoo Information Systems Company, Korea  
jeuk@disc.co.kr

**Abstract.** This paper presents a variant of the AdaBoost algorithm for boosting Naïve Bayes text classifier, called AdaBUS, which combines active learning with boosting algorithm. Boosting has been evaluated to effectively improve the accuracy of machine-learning based classifiers. However, Naïve Bayes classifier, which is remarkably successful in practice for text classification problems, is known not to work well with the boosting technique due to its instability of base classifiers. The proposed algorithm focuses on boosting Naïve Bayes classifiers by performing active learning at each iteration of boosting process. The basic idea is to induce perturbation of base classifiers by augmenting the training set with the most informative unlabeled documents.

**Keywords:** Text classification, Naïve Bayes, boosting, active learning, selective sampling

## 1 Introduction

Boosting is a general technique for improving the accuracy of machine-learning based classifiers, which combines a series of base (or weak) classifiers to produce a single powerful classifier. Since the boosting technique was developed by Yoav Freund and Rob Schapire [4], it has been considered to be one of the best approach to improving classifiers in many previous studies [13]. In particular, boosting contributes to significantly improve the decision tree learning algorithm [5, 6, 11].

In our work, we focus on boosting Naïve Bayes classifier, which is a simple yet surprisingly accurate technique and has been used in many classification projects [1]. Particularly, for text classification problems, Naïve Bayes classifier is known to be remarkably successful in practice, despite the fact that text data generally has a huge number of attributes (features) [10]. For this reason, it is very significant to apply boosting to Naïve Bayes text classifiers. However, Naïve Bayes classifier is known to perform poorly with boosting [14]. This is because effective boosting, in principle, requires high variance (or instability) in

---

<sup>\*</sup> This research was supported by University of Seoul, Korea, in the year of 2003.

the accuracy of base classifiers [14], but Naïve Bayes is relatively stable with respect to changes of training set.

This paper presents a variant of the AdaBoost algorithm for boosting Naïve Bayes text classifier, called AdaBUS, which combines active learning with the AdaBoost boosting algorithm. In active learning approach, learner automatically selects the most informative examples for class labeling and training, without depending on a teacher's decision or random sampling. To this end, we use uncertainty-based selective sampling method [7, 8], which has been frequently used for learning with text data. For selective sampling, we propose a uncertainty measure fit for Naïve Bayes learning framework.

## 2 Preliminaries

### 2.1 Naïve Bayes Learning Framework for Text Classification

Learning a Naïve Bayes (NB) text classifier means estimating the parameters of generative model by using a set of labeled training documents. The estimated classification model is composed of two parameter: the word probability estimates  $\hat{\theta}_{w|c}$ , and the class prior probabilities  $\hat{\theta}_c$ ; that is, classification model  $\hat{\theta}_{NB} = \{\hat{\theta}_{w|c}, \hat{\theta}_c\}$ . Each parameter can be estimated according to maximum a posteriori (MAP) estimation<sup>1</sup>.

For classifying a given document, Naïve Bayes learning method estimates the posterior probability of a class via Bayes' rule; that is,  $Pr(c_j|d_i) = \frac{Pr(c_j) \cdot Pr(d_i|c_j)}{Pr(d_i)}$ , where  $Pr(c_j)$  is the class prior probability that any random document from the document collection belongs to the class  $c_j$ ,  $Pr(d_i|c_j)$  is the probability that a randomly chosen document from documents in the class  $c_j$  is the document  $d_i$ , and  $Pr(d_i)$  is the probability that a randomly chosen document from the whole collection is the document  $d_i$ . The document  $d_i$  is then assigned to a class  $\arg\max_{c_j \in C} Pr(c_j|d_i)$  with the most posterior<sup>2</sup>. Here, the document  $d_i$  is represented by a bag of words  $(w_{i1}, w_{i2}, \dots, w_{i|d_i|})$  where multiple occurrences of words are preserved. Moreover, Naïve Bayes classifier is based on the simplifying assumption that the terms in a document are mutually independent and the probability of term occurrence is independent of position within the document. This assumption results in the following classification function  $f_{\hat{\theta}_{NB}}(d_i) = \arg\max_{c_j \in C} Pr(c_j|d_i) = \arg\max_{c_j \in C} Pr(c_j) \cdot \prod_{k=1}^{|d_i|} Pr(w_{ik}|c_j)$ .

To generate this classification function,  $Pr(c_j)$  can be simply estimated by counting the frequency with which each class value  $c_j$  occurs in a set of the training documents  $D^t$ , where  $Pr(c_j|d_i) \in \{0, 1\}$ , given by the class label. That is,  $Pr(c_j) = \hat{\theta}_{c_j} = \frac{\sum_{i=1}^{|D^t|} Pr(c_j|d_i)}{|D^t|}$ . As for  $Pr(w_{ik}|c_j)$ , its maximum likelihood estimate using Laplace's law of succession [9] is  $Pr(w_{ik}|c_j) = \hat{\theta}_{w_{ik}|c_j} =$

<sup>1</sup> The MAP estimation is to find maximally probable model  $\theta_{MAP}$  among possible models  $\Theta$ , given a set of training documents  $D$ ; that is,  $\theta_{MAP} \equiv \arg\max_{\theta \in \Theta} Pr(\theta|D) = \arg\max_{\theta \in \Theta} \frac{Pr(D|\theta) \cdot Pr(\theta)}{Pr(D)} = \arg\max_{\theta \in \Theta} Pr(D|\theta) \cdot Pr(\theta)$ .

<sup>2</sup>  $\arg\max_{x \in X} f(x)$  is the value of  $x$  that maximizes  $f(x)$ .

**Table 1.** The AdaBoost Algorithm with Naïve Bayes.

---

Input: A set of training documents $D^t = \{\langle d_i, c_j \rangle   d_i \in D, c_j \in C\}$
Output: A classifier $f_{\hat{\theta}_{NB}}$ where $\hat{\theta}_{NB} = \{\hat{\theta}_{w c}, \hat{\theta}_c\}$

---


$$l_{NB}(D^{tl}) = \operatorname{argmax}_{\hat{\theta}_{NB}} Pr(D^t | \hat{\theta}_{NB}) \cdot Pr(\hat{\theta}_{NB}) \text{ /* MAP estimation */}$$

$$f_{\theta_{NB}}(d_i) = \operatorname{argmax}_{c_j \in C} Pr(c_j | d_i) = \operatorname{argmax}_{c_j \in C} Pr(c_j) \cdot \prod_{k=1}^{|d_i|} Pr(w_{ik} | c_j)$$

$$\mathbf{w}^{(t)} = (w_1^{(t)}, \dots, w_{|D^t|}^{(t)}) \text{ /* weight distribution */}$$


---

1. **Initialize:**  $w_{d_i}^{(1)} = \frac{1}{|D^t|}$  for all  $d_i \in D^t$
2. **Learn:** repeat for  $t = 1$  to  $T$ 
  - (a) Estimate a classification model with respect to the weighted training documents  
 $\hat{\theta}_{NB} = l_{NB}(D^t)$
  - (b) Build a base classifier  $f_{\hat{\theta}_{NB}}$  with the estimated model  $\hat{\theta}_{NB}$
  - (c) Calculate the weighted training error  $\epsilon_t$  of  $\hat{\theta}_{NB}$   

$$\epsilon_t = \sum_{d_i \in D^t} \left[ w_{d_i}^{(t)} \cdot \mathbf{I} \left( f_{\hat{\theta}_{NB}}^{(t)}(d_i) \neq f_{\theta}(d_i) \right) \right]$$

/\*  $\mathbf{I}(x) = 1$  if  $x$  is true, 0 otherwise. \*/
  - (d) Calculate confidence  $\alpha_t$  of  $\hat{\theta}_{NB}$ :  

$$\alpha_t = \frac{1}{2} \ln \left( \frac{1 - \epsilon_t}{\epsilon_t} \right)$$
  - (e) Update weights for the next iteration:  

$$w_{d_i}^{(t+1)} = \frac{w_{d_i}^{(t)}}{z_t} \times \begin{cases} e^{-\alpha_t} & \text{if } f_{\hat{\theta}_{NB}}^{(t)}(d_i) = f_{\theta}(d_i) \\ e^{+\alpha_t} & \text{if } f_{\hat{\theta}_{NB}}^{(t)}(d_i) \neq f_{\theta}(d_i) \end{cases}$$

where  $z_t$  is a normalization factor so that  $\mathbf{w}^{(t)}$  is a probabilistic distribution,  
i.e.,  $\sum_{d_i \in D^t} w_{d_i}^{(t)} = 1$
3. **Return the final classifier:**
  - (a) weighted majority voting of the generated classifiers  $\{f_{\hat{\theta}_{NB}}^{(t)}\}_{t=1}^T$ :  

$$f_{\hat{\theta}_{NB}}(d) = \operatorname{argmax}_{c_i \in C} \sum_{t=1}^T \left[ \frac{\alpha_t}{\sum_{r=1}^T \alpha_r} \cdot \mathbf{I} \left( f_{\hat{\theta}_{NB}}^{(t)}(d) = f_{\theta}(d) \right) \right]$$

---

$\frac{tf_{c_j}(w_{ik})+1}{\sum_{w \in V} tf_{c_j}(w)+|V|}$ , where  $tf_{c_j}(w)$  is the number of occurrences of the word  $w$  in the class  $c_j$  and  $V$  denotes the set of significant words extracted from the training documents.

## 2.2 The AdaBoost Boosting Algorithm with Naïve Bayes

This section presents the procedure of AdaBoost algorithm with Naïve Bayes learning, which is described by using notations introduced in the previous section and Naïve Bayes learning framework. As shown in Table 1, the boosting algorithm is composed of two phase: the *learning* phase and the *voting* phase.

The *learning* phase induces a series of base classifiers while repeatedly updating the distribution of weights of training examples, based on previously generated base classifiers. Initially, the boosting process sets the initial weight of

each of training documents to be  $\frac{1}{|D^t|}$  (see line 1). At each iteration, the process estimates a classification model  $\hat{\theta}_{NB}$  with respect to the weighted training documents, and then generates a base classifier using the estimated model  $\hat{\theta}_{NB}$  as shown in lines 2(a)-(b). After a base classifier is learned, the weights of training examples are individually modified to allow the subsequent classifier to focus on misclassified training examples. For this, the weighted training error  $\epsilon_t$  for the estimated model  $\hat{\theta}_{NB}$  is calculated as shown in line 2(c)<sup>3</sup>, and then, with the error  $\epsilon_t$ , the confidence  $\alpha_t$  that measures the importance of  $\hat{\theta}_{NB}$  is computed. This measure gets larger as  $\epsilon_t$  gets smaller, as shown in line 2(d). After that, the confidence measure  $\alpha_t$  is used for re-weighting training documents as shown in line 2(e); that is, the process increases the weight of documents incorrectly classified by the currently estimated model  $\hat{\theta}_{NB}$  whereas it decreases the weight of correctly classified documents.

After performing  $T$  rounds, in the *voting* phase, the process combines the  $T$  multiple base classifiers into an improved classifier through voting. As shown in line 3, the boosted final classifier returned is a weighted majority vote of the base classifiers, in which the confidence  $\alpha_t$  is re-used for voting with the weight corresponding to its value.

### 3 Boosting Naïve Bayes

#### 3.1 Basic Idea

As mentioned before, to achieve effective boosting of Naïve Bayes, we should find how to increase the instability of base classifiers. Our strategy for boosting Naïve Bayes is an incremental augmentation of a set of training documents through *active learning* approach. In active learning approach, learner actively chooses the informative training documents from a pool of unlabeled documents with the aim of reducing the number of training examples while maintaining high classification accuracy [2].

At each iteration of boosting process, well chosen informative documents are added to the current training set according to the principle of active learning. By learning with the augmented training set different from the previous one, a newly generated classification model is expected not to be similar to the previous model within a set of all possible models. This perturbation of training set can introduce instability into Naïve Bayes classifiers. Besides, since the training set is augmented with highly informative training examples, a base classifier built by learning with such examples is expected to be gradually improved at each iteration. The proposed strategy can prevent an increase in classification error of the base classifiers probably incurred by the intended instability. Consequently, the base classifiers with higher instability and higher accuracy are combined into a more powerful boosted classifier.

<sup>3</sup> The equation  $f_{\theta}(d) = c_j$  represents that the human labeler (who is assumed to know the true classification model  $\theta$ ) determines the true class label of a given document  $d_i$  to be  $c_j$ .

**Table 2.** Modification to the AdaBoost Algorithm with Naïve Bayes: AdaBUS.

---

Input: A set of training documents $D^t = \{\langle d_i, c_j \rangle   d_i \in D, c_j \in C\}$
A set of unlabeled documents $D^u = \{d_k   d_k \in D\}$
Output: A classifier $f_{\hat{\theta}_{NB}}$ where $\hat{\theta}_{NB} = \{\hat{\theta}_{w c}, \hat{\theta}_c\}$

---

1. **Initialize:**  $w_{d_i}^{(1)} = \frac{1}{|D^t|}$  for all  $d_i \in D^t$
2. **Learn:** repeat for  $t = 1$  to  $T$ 
  - (a) Estimate a base classifier with respect to the weighted training documents  
 $\hat{\theta}_{NB} = l_{NB}(D^t)$
  - (b) Build a base classifier  $f_{\hat{\theta}_{NB}}$  with the estimated model  $\hat{\theta}_{NB}$
  - (c) Calculate the weighted training error  $\epsilon_t$  of  $\hat{\theta}_{NB}$   

$$\epsilon_t = \sum_{d_i \in D^t} \left[ w_{d_i}^{(t)} \cdot \mathbf{I} \left( f_{\hat{\theta}_{NB}}^{(t)}(d_i) \neq f_{\theta}(d_i) \right) \right]$$
  - (d) Calculate confidence  $\alpha_t$  of  $\hat{\theta}_{NB}$ :  

$$\alpha_t = \frac{1}{2} \ln \left( \frac{1 - \epsilon_t}{\epsilon_t} \right)$$
  - (e) Find a document with the largest uncertainty from unlabeled documents:  
 $d_{max} = \operatorname{argmax}_{\mathbf{d} \in D^u} \text{CU}_{NB}(\mathbf{d})$
  - (f) Break if  $\text{CU}(d_{max}) < \mu$   
 $\quad \quad \quad / * \mu$  is a threshold value for selective sampling  $/*$
  - (h) Augment the current training set by labeling the selected document:  
 $D^t = D^t \cup \langle d_{max}, f_{\hat{\theta}_{NB}}(d_{max}) \rangle$
  - (g) Determine the initial weight of the selected document:  

$$w_{d_{max}} = \frac{\sum_{d_i \in D^t} w_{d_i}^{(t)}}{|D^t|}$$
  - (i) Update weights:  

$$w_{d_i}^{(t+1)} = \frac{w_{d_i}^{(t)}}{z_t} \times \begin{cases} e^{-\alpha_t} & \text{if } f_{\hat{\theta}_{NB}}^{(t)}(d_i) = f_{\theta}(d_i) \\ e^{+\alpha_t} & \text{if } f_{\hat{\theta}_{NB}}^{(t)}(d_i) \neq f_{\theta}(d_i) \end{cases}$$
3. **Return the final classifier:**
  - (a) weighted majority voting of the generated classifiers  $\{f_{\hat{\theta}_{NB}}^{(t)}\}_{t=1}^T$ :  

$$f_{\hat{\theta}_{NB}}(\mathbf{d}) = \operatorname{argmax}_{c_i \in C} \sum_{t=1}^T \left[ \frac{\alpha_t}{\sum_{r=1}^T \alpha_r} \cdot \mathbf{I} \left( f_{\hat{\theta}_{NB}}^{(t)}(\mathbf{d}) = f_{\theta}(\mathbf{d}) \right) \right]$$

---

The challenging problem is how to isolate the best candidate training examples from a set of unlabeled document set. In our work, we adopt the uncertainty-based selective sampling [7] because it has been frequently used for learning with text data. The sampling process is performed based on classification uncertainty which is the degree of uncertainty in the classification of a test example with respect to the currently derived model. In the following subsection, an uncertainty measure fit for Naïve Bayes learning method is devised.

### 3.2 Adaptive Boosting with Uncertainty-Based Selective Sampling: AdaBUS

**Uncertainty-Based Selective Sampling.** As we have already seen, Naïve Bayes learning method develops the probability distribution over words  $W$  for



each given class  $c$  (i.e.,  $Pr(W|c)$ ) that accounts for the concept of that class. In this regard, if a document's classification is *uncertain* under the current model, we can say that the word distribution for its correct class is still not well developed for classification. In such case, the probability distribution over words occurring in the input document for its correct class is similar (or near) to those of other classes. From this, we find that the classification uncertainty can be determined by measuring the distances between the word distributions learned. Now, we propose an uncertainty measure based on the Kullback-Leibler (KL) divergence, which is a standard information-theoretic measure of distance between two probability mass functions [3]. For a document  $d$ , its KL divergence between the word distributions induced by the two classes  $c_i$  and  $c_j$ ,  $KLdist_d(Pr(W|c_i), Pr(W|c_j))$  is defined as  $\sum_{w_k \in d} Pr(w_k|c_i) \cdot \log \frac{Pr(w_k|c_i)}{Pr(w_k|c_j)}$ . Its classification uncertainty  $CU(d)$  is defined as follows:

$$CU(d) = 1 - \frac{\sum_{c_i, c_j \in C} KLdist_d(Pr(W|c_i), Pr(W|c_j))}{|C| \cdot (|C| - 1)} \quad (1)$$

where  $|C|$  denotes the total number of the existing classes. Note that the value of  $KLdist_d(\cdot)$  is measured, not over all words but over only those words belonging to a categorized document  $d$ .

**Incorporating Active Learning into AdaBoost.** With the proposed uncertainty measure, tasks for incremental updates of the training set are performed in the conventional boosting process. Table 2 shows modification to the AdaBoost algorithm that includes the additional selective sampling tasks. The lines 2(e)-(h) corresponding to the selective sampling task are added to the Table 1 of AdaBoost algorithm.

As shown in line 2(e), the process finds a document with the largest uncertainty from a given unlabeled documents  $D^u$ . The selected document  $d_{max}$  is then checked whether its classification uncertainty is larger than a given threshold value  $\mu$  (see line 2(f)). If so, its appropriate class label is assigned to the document by the human labeler, and moreover the initial weight of  $d_{max}$  is set to the average of current weights of training examples (see lines 2(g)-(h)). The initially determined weight will be modified according to the re-weighting rule.

In our algorithm, the optimal time when to stop the process is determined through tuning the threshold value of uncertainty. By doing so, we can adaptively determine the termination time according to the status of current learning process. The reason why the termination policy is that the selected documents' low uncertainty suggests that further iterations will not generate any more refined base classifiers different from previous one. If the training set is sufficiently augmented, the base classifier will probably become stable without further improvement.

**Table 3.** Setup of test document sets.

Category	Total number of documents	Number of initial training set	Number of test document set	Number of unlabeled training set
acq	264	5	30	229
crude	165	5	30	130
earn	504	5	30	469
interest	249	5	30	214
money-fx	179	5	30	144
ship	222	5	30	187
trade	150	5	30	115

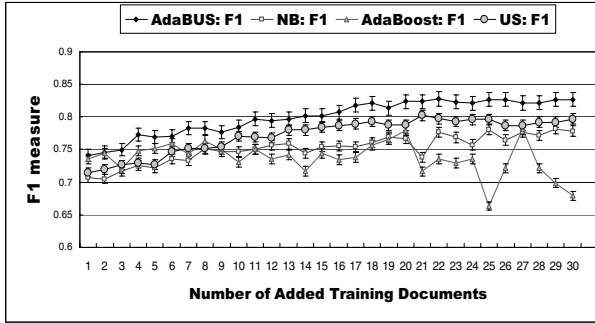
## 4 Experimental Results

### 4.1 Experimental Setup

In order to evaluate our method, we used the Reuters-21578 SGML document collection [12] which has been most commonly used in the text classification literature. This data set consists of 21,578 articles, each one pre-labeled with one or more of 135 topics (categories). However, the Reuters collection has a very skewed distribution of documents over 135 categories, and thus it has often been criticized as a poor collection for classification. For a more reliable evaluation, we generated a subset of the Reuters collection in which documents are not skewed over categories. First, we selected the documents belonging to the 7 most frequent categories including ‘acq’, ‘crude’, ‘earn’, ‘interest’, ‘money-fx’, ‘ship’, ‘trade’. Among those selected documents, we again chose 1,733 documents that had a single category in order to avoid the ambiguity of documents with multiple topics. This controlled document sets are described in detail in Table 3. In our experiment, the simulation results are discussed with respect to the F1-measure, which gives equal weight to both recall and precision. This measure varies from 0 to 1, and are proportionally related to classification effectiveness. With the F1-measure for each category, we compute the overall F1-measure for the collection by macro-averaging.

### 4.2 Performance Analysis

**Effect of Selective Sampling in Boosting Process.** Figure 1 illustrates the changes in F1-measure according to the number of training documents added to the initial training set. Note that the conventional AdaBoost algorithm without active learning does not require augmenting the initial training set, unlike the proposed method (denoted as AdaBUS) and uncertainty-based sampling method (denoted as US). Thus, in case of AdaBoost and NB, the total number of training documents should be equal to the sum of the number of the initial training documents and the number of appended training documents (which is represented in horizontal axis of Figure 1). Therefore, all the methods suggested in our experiment require the same amount of human effort.



**Fig. 1.** Changes in F1-measure from varying the number of appended training documents: AdaBUS denotes the proposed method, NB Naïve Bayes without boosting, AdaBoost the conventional AdaBoost algorithm, and US uncertainty-based selective sampling method.

As shown in this figure, we have observed that the proposed AdaBUS method is successful in boosting Naïve Bayes; AdaBUS could increase the quality of Naïve Bayes classifier with an average increase of 10% in the F1-measure over pure Naïve Bayes algorithm (NB).

As for the AdaBoost algorithm, [14] described that its boosting process cannot increase the quality of Naïve Bayes classifier. Even, in our experiment, the AdaBoost algorithm is worse than pure Naïve Bayes in most cases, as shown in Figure 1. This is probably because each of base classifiers can have a lower accuracy<sup>4</sup> through learning with an insufficiently number of training examples. In addition, our experiment includes a comparison with uncertainty-based sampling method (denoted as US) because we cannot exclude the possibility that only the sampling method outperforms AdaBUS method. Fortunately, we can observe that AdaBUS method gives benefit over the sampling method. However, note that uncertainty-based sampling improves the accuracy of Naïve Bayes classifier, as shown in Figure 1, since it can allow to compose training set with the most informative documents from a pool of unlabeled documents. In short, the proposed method can make it possible to increase variance and simultaneously to increase F1-measure of base classifiers in Naïve Bayes learning, which results in effectively boosting Naïve Bayes classifier.

## 5 Conclusions and Future Work

We have presented a method for boosting Naïve Bayes text classifiers by using active learning (i.e., uncertainty-based selective sampling) approach. The basic idea behind our approach is to increase the variance in base classifiers by incrementally augmenting a set of training documents with selectively sampled

<sup>4</sup> Basically, in case of binary classifier, effective boosting requires that the accuracy of base classifier is larger than 0.5.

documents. To this end, we propose a special uncertainty measure fit for Naïve Bayes learning. In the future, we plan to combine EM algorithm with AdaBUS algorithm considering the fact that EM process can further improves the base classifiers without additional human efforts.

## References

1. R. Aggrawal, R.J. Bayardo, and R. Srikant, "Athena: Mining-based Interactive Management of Text Databases," *Proceedings of the 7<sup>th</sup> International Conference on Extending Database Technology*, pp. 365–379, 2000.
2. S. A.-Engelson, and I. Dagan, "Committee-Based Sample Selection for Probabilistic Classifiers," *Journal of Artificial Intelligence Research*, Vol. 11, pp. 335–360, 1999.
3. T.M. Cover, and J.A. Thomas, *Elements of Information Theory*, Wiley, 1991.
4. Y. Freund, R.E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Proceedings of the 2<sup>th</sup> European Conference on Computational Learning Theory*, 1995.
5. Y. Freund and R.E. Schapire, "Experiments with a New Boosting Algorithm," *International Conference on Machine Learning*, pp. 148–156, 1996.
6. J.H. Friedman, T. Hastie, and R. Tibshirani, "Additive logistic regression: A statistical view of boosting," *Annals of Statistics*, Vol.28, No.2, pp. 337–374, 2000.
7. D.D. Lewis, and J. Catlett, "Heterogeneous Uncertainty Sampling for Supervised Learning," *Proceedings of the 11<sup>th</sup> international Conference on Machine Learning*, pp. 148–156, 1994.
8. M. Lindenbaum, S. Markovitch, and D. Rusakov, "Selective sampling for nearest neighbor classifiers," *American Association for Artificial Intelligence*, 1999.
9. T.M. Mitchell, "Bayesian Learning," *Machine Learning*, McGraw-Hill, pp. 154–200, 1997.
10. K. Nigam, A. McCallum, S. Thrun, and T.M. Mitchell, "Learning to Classify Text from Labeled and Unlabeled Documents," *Proceedings of the 15<sup>th</sup> National Conference on Artificial Intelligence and the 10<sup>th</sup> Conference on Innovative Applications of Artificial Intelligence*, pp. 792–799, 1998.
11. J.R. Quinlan, "Bagging, boosting, and C4.5," *Proceedings of the 13<sup>th</sup> National Conference on Artificial Intelligence*, pp. 725–730, 1996.
12. D.D Lewis, "Reuters-21578 text categorization test collection," <http://kdd.ics.uci.edu/databases/reuters21578/>, 1997.
13. R.E. Schapire, and Y. Singer, "BoosTexter: A Boosting-based System for Text Categorization," *Machine Learning*, Vol.39, No.2, pp. 135–168, 2000.
14. K.M. Ting, and Z. Zheng, "A study of AdaBoost with Naïve Bayesian Classifiers: Weakness and Improvement," *Computational Intelligence*, Vol.19, No.2, pp.186–200, 2003.

# Using Interval Association Rules to Identify Dubious Data Values

Ren Lu, Mong Li Lee, and Wynne Hsu

Department of Computer Science, National University of Singapore  
`iscp1247@nus.edu.sg`, `{leeml,whsu}@comp.nus.edu.sg`

**Abstract.** A hard-to-catch erroneous data is one whose value looks perfectly legitimate. Yet, if we examine this value in conjunction with other attribute values, the value appear questionable. Detecting such dubious values is a major problem in data cleaning. This paper presents a framework to automatically detect dubious data values in the datasets. Data is first pre-processed by data smoothing and mapping. Next, interval association rules are generated which involved data partitioning via clustering before the rules are generated using an Apriori algorithm. Finally, these rules are used to identify data values that fall outside the expected intervals. Experiment results show that the proposed framework is able to accurately and efficiently dubious values in large datasets.

## 1 Introduction

The globalization of many organizations has led to the generation of large amounts of data that are obtained from different geographical locations. The quality of these data is critical for many industrial applications [12] given that many decisions are made based on the information derived from the data.

There are many causes of dirty data, ranging from data entry errors, variants of abbreviations and spelling, missing values, duplicate records, to unit differences. Data cleaning deals with the detection and removal of errors and inconsistencies from data to improve the quality of data [6]. In general, data cleaning processes aims to standardize format, remove anomalies, infer missing values, rectify errors such as outlier and out-of-range values, and eliminate duplicates.

Much research efforts have been concentrated on the issue of duplicate elimination [3, 2]. The detection of outliers has been studied in statistics community [16]. However, most statistics methods require the prior knowledge of data distribution which may not be possible in real-life applications. Another approach to solve the problem of outliers is to apply data mining techniques. However, the techniques developed in [7, 8] are limited to date fields. [1] present a pattern identification approach for deviation detection. [5, 9, 14, 4, 15] utilize clustering algorithms to detect outliers. In all these techniques, correlations among a subset of attribute values are ignored.

In this paper, we introduce the notion of dubious data. Dubious data refers to data values that appear legitimate on their own. However, upon examination together with other corresponding attribute values, their values become

questionable. For example, an assessment grade of “D” is perfectly legitimate. However, if we also know that the corresponding examination and test scores are all in the range of “A+”, this grade of “D” becomes highly suspicious. In another example, we consider the customer database of an insurance company (see Table 1). At first glance, all the data seems correct. However, on closer examination, we note that income value for Record 2 is dubious because it does not fall in the expected range of income [25,000, 30,000] that managers generally earn. This could be the result of a typographical error, and the correct income should probably be 26,000. We observe that dubious data values can be detected via data correlation, which measures the relationship between associated variables. Such correlations can be captured by using interval association rules.

In this paper, we propose a framework that uses interval association rules to identify dubious data. We extend the algorithm in [14] to include both equality predicates ( $Attr = val$ ) as well as range predicates ( $Attr = [val1, val2]$ ). Experiment results demonstrate the accuracy and efficiency of our proposed approach.

## 2 Related Works

Most existing work in the detection of outliers and deviation analysis lies in the field of statistics [16] (e.g. mean and variance, upper or lower outliers in an ordered sample). [7] also employs statistics measures such as averages, standard deviation, and range to identify outliers. However, the main problem with these approaches is that in a number of situations, the user simply does not have enough knowledge about the underlying data distribution [14].

Outlier detection is also an important issue in clustering. [5, 7, 9, 14, 15] present density-based, hierarchical and distance-based clustering methods to detect outliers. However, most of them are only consider the numeric data. However, they regard the entire data record as a whole and do not consider the deviation of one attribute value with respect to other attribute values. [4] proposes an outlier detection algorithm to find abnormal data on subspace by using lower dimensional projections. For the algorithm to work, it has to assume that the data is uniformly distributed which, in general, is not true.

[1] puts forward a linear method that identifies deviation after observing a series of similar data. However, the deviation to be detected largely depends on the form of the chosen pattern and the technique in [1]. It does not handle dubious data. [7] proposes another pattern identification method. The patterns in [7] refer to a set of records with the same empty fields. These records are formed by partitioning. Records that deviate from the discovered patterns are identified as outliers. Once again, this approach does not handle dubious data.

[8] utilizes ordinal association rules such as  $i_1, i_2 \Rightarrow i_1 \text{ op } i_2$ , where  $\text{op} \in \{\leq, \geq, =\}$  to find records that do not conform to the rules. However, ordinal association rule in [8] is only applicable to date field data.

[13] introduces the notion of quantitative association rules. It partitions numeric data into equi-depth interval and then generate rules based on these intervals. Unfortunately, using equi-depth may not give us interesting or meaningful

**Table 1.** Example of Dubious Data Values

Name	Sex	Job	Income	Date
Jack	M	Lawyer	45,000	2003-01-10
Mary	M	Manager	6,000	2003-01-20
Tom	M	Lawyer	50,000	2003-01-09
Jane	F	Manager	30,000	2003-01-25
Dorothy	F	Manager	25,000	2003-01-22

intervals. [11] employs clustering techniques to obtain better quality partitions of numeric data. In this paper, we extend the clustering method in [11] to categorical data for dubious data detection.

### 3 Proposed Framework

We proposed a framework for the detection of dubious data. We first select a subset of the dataset as our training data for learning the significant data correlations that exist in the dataset. To reduce the influence of noise on this training dataset, we apply data smoothing operations. This involves inferring missing values in the dataset and standardizing attribute value format. Categorical data and date values are also mapped to ordinal and numeric values to facilitate subsequent comparisons. Next, we build interval association rules from these data. This is carried out as follows:

1. Partition the data into meaningful segments based on clusters. A meaningful segment implies similar data density within the segment.
2. Generate interval association rules using some Apriori [10] algorithms.

Finally, we apply the interval association rules to the dataset to be cleaned. For each tuple in the dataset, we check if it is covered by some rules. A tuple may be covered by multiple rules. In this case, we keep all the rules for determining the expected value range of the tuple. On the other hand, it is also possible that none of the rules cover the tuple. When this happens, we select the rule with the highest confidence measures. For example:

**Rule1:** Sex = 2  $\rightarrow$  Income = [45,000, 50,000] (conf: 66.7% sup: 60%)

**Rule2:** Job = 2, Date = [20, 25]  $\rightarrow$  Income = [25,000, 30,000] (conf: 66.7%, sup: 60%)

**Rule3:** Job = 1, Date = [9, 10]  $\rightarrow$  Income = [45,000, 50,000] (conf: 100%)

**Record1:** Job = Manager, Income = 45,000 Date = 2003-01-22

**Record2:** Job = Layer, Income = 40,000, Date = 2003-01-02

After preprocessing, we discover that Record1 matches Rule1 and Rule2. Hence, both Rule1 and Rule2 are kept for subsequent dubious data detection. Meanwhile, Record2 is not covered by any of the rules. In this case, we select the highest confidence rule, Rule3, to cover Record2. At the end of this phase, each tuple is covered by at least one rule. Next, we compare the observed value of the tuple with that predicted by the covered rules. If the value of the tuple deviates from the predicted value of rules, we consider the current tuple as a dubious data that requires further checking.

### 3.1 Preprocessing

A real-world dataset often contains noise such as missing data, varying data format etc. Such noise may interfere with the capturing of the underlying data values association. In order to build a more representative association rule model, we first perform the following preprocessing operations.

1. **Infer missing values.** Multiple imputation is one of the most common methods used to infer missing values. This techniques replaces each missing value with a set of plausible values that represent the uncertainty about the correct value to assign. These multiple imputed data sets are then analyzed to complete the data. Note that there is no single “one size fits all” solution to solve incomplete data problems. In fact, filling missing values in a dataset remains an empirical work that is dependent on the data characteristics.
2. **Standardize data format.** Attributes such as date can have varying formats such as “YYYY/MM/DD” or “DD-MM-YYYY”. One of the these formats is adopted as the standard for the dataset, and the attribute values are converted to conform to this format.
3. **Mapping.** Next, we transform the data values of categorical and date attributes to facilitate the subsequent comparison of data values. Categorical data is a common data type which occurs frequently as a string of characters. In order to simplify string comparison between categorical data values, we map the values of each categorical attribute to ordinal values. To handle date attributes efficiently, we pre-determine a start date, and transform the date values in the dataset to a value that is equal to the number of days between now and the start date. Table 2 shows the resulting table after a series of mappings has been applied to Table 1.

**Table 2.** Result of Table 1 after Mapping Operations

Manager  $\xrightarrow{map}$  2, Lawyer  $\xrightarrow{map}$  1  
Start day = 2003-01-01

Name	Sex	Job	Income	Date
1	1	1	45,000	10
2	1	2	6,000	20
3	1	1	50,000	9
4	2	2	30,000	25
5	2	2	25,000	22

### 3.2 Generating Interval Association Rules

**Step 1. Partitioning of Data.** In many real-life applications, a dubious data value often refer to the situation when a data value falls outside of an acceptable range. To determine what is an acceptable or expected range for each data



value when it appears together with other attribute values, we generate interval association rules. This is achieved by first dividing the data values into segments based on clustering techniques which divide data into the groups of similar objects. Therefore, the segments derived from clustering are relatively meaningful under the whole distribution. An Apriori-based association rules discovery algorithm is then applied to these segments to form the interval association rules.

Let us first extend the clustering technique proposed in [11] which use the distances between data points to cluster both numeric and categorical data values. It is worthwhile to note that [15] develops a clustering approach that enables clustering on the whole dataset with both continuous and categorical attributes. However, in our case, we need to cluster with respect to a particular attribute. Thus, in our extension, we propose distance metrics that can give us an intuitive measurement of how close is the input data to a cluster with respect to a particular attribute.

**Definition 1.**  $C_X$  is the cluster projected on the attribute  $X$  ( $X$  is a particular attribute of certain database  $B$ ).  $C_X$  has to satisfy the following conditions:

1.  $d_{C_X} \leq$  threshold density  $d_n$  if  $X$  is a numeric attribute;  
 $d_{C_X} \leq$  threshold density  $d_c$  if  $X$  is a categorical attribute.
2.  $|C_X| \geq$  frequent bound  $S_0$ .

In the definition 1,  $d_{C_X}$  represents the diameter of the cluster.  $|C_X|$  is the number of points in the cluster.

Figure 2 illustrates the above definition. It shows the clusters over two numeric attributes *Income* and *Date* obtained from the dataset as reported in Table 2.  $C_{Income}$  is the cluster projected on *Income* while  $C_{Date}$  is the cluster on *Date*.

In general, we observe that users are interested in each category of a categorical attribute. Therefore, usually we have  $d_c$  equal to zero. However, we can enlarge  $d_c$  if taxonomy of classes exists and group the data under the same class. This process is somewhat similar to association rules generalization.

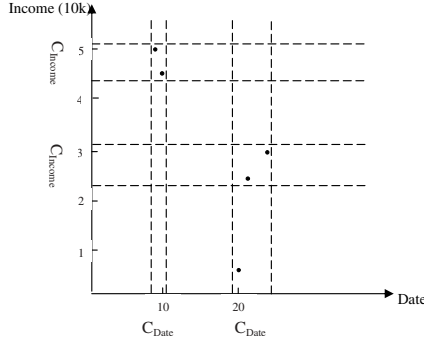
For numeric attributes, in order to partition the data into proper intervals, each numeric attribute will have its own diameter threshold.

To clustering data, suppose  $x_{ij}$ , the value of the attribute  $j$  of the  $i$ th tuple. It is inserted to the closest cluster  $C_j$ , which is defined as the cluster that minimizes a given inter-group distance metric ( $d = \frac{\sum_{k=1}^N \delta(x_{ij}, x_{kj})}{N}$ ,  $x_{kj}$  is the point in  $C_j$  and  $N$  is the number of points in  $C_j$ ). For closest cluster entry, it means its  $d$  is the minimum for all the clusters on attribute  $j$ .  $\delta(x_{ij}, x_{kj})$  is the distance metrics for two points. In our extension, the function to measure the distance between a pair of points is different for numeric and categorical attributes. We choose two general forms of distance metrics for these two kinds of attributes.

For numeric data, there are many functions to compute the distance between two points like Euclidean distance, Manhattan distance and Minkowski distance etc. In our experiments, we use the Manhattan distance function given by:

$$d1 = |x_{ij} - x_{kj}| \quad (1)$$

where  $x_{ij}$  and  $x_{kj}$  represent the values of the  $i$ th and  $k$ th records on attribute  $j$ .



**Fig. 1.** Clusters over Income and Date attributes

For categorical data, we use the Sigma function as defined below:

$$d2 = \begin{cases} 0 & x_{ij} = x_{kj} \\ 1 & x_{ij} \neq x_{kj} \end{cases} \quad (2)$$

The use of Sigma function is possible due to the mappings that have been carried out to convert string values of categorical attributes to ordinal values.

We present a formula to combines  $d1$  and  $d2$  simultaneously:

$$\delta(x_{ij}, x_{kj}) = w_j * d1 + (1 - w_j) * d2 \quad (3)$$

$w_j$  denotes a dummy value on the attribute  $j$ , if the data is numeric data,  $w_j = 1$ ,  $d1$  is chosen; otherwise, if the data belongs to categorical attributes,  $w_j = 0$ , then  $d2$  is selected.

If the cluster diameter is greater than the threshold, then we split the cluster by choosing the farthest two points. Point in the current cluster will be redistributed to the new clusters based on the closest distance. After all the clusters are built, the data segments are obtained from these clusters by setting the upper and lower bound with two points which are farthest away in the cluster.

**Step 2. Rule Generation.** With the formation of data segments, we now consider these data segments to be the members of size-1 frequent itemsets. Interval association rules can then be generated based on these data segments. [14] presents a distance-based association rule algorithm. However, their solution can only work well under a high distance threshold implying a high confidence in classical association rule. This may lead to some useful rules with lower confidence being missed. Incomplete rule set may increase false alarms during dubious data identification. Here, we adopt the use of the existing Apriori algorithm [10] to build association rules. The reason is that Apriori algorithm is by far the most well-known and basic association algorithm. It can find all large itemsets to build desired rules in terms of *minsup* and *minconf*. Now, each interval can

be viewed as a discrete value. Candidate generation and counting are obtained based on these data intervals. Figure 4 shows the frequent items with minconf at 100%, minsup at 40% and the corresponding samples of the interval association rules generated.

For our purpose of dubious data detection, we focus on the set of interval association rules whose right hand side contain only one target attribute. This allows us to focus on one target attribute at a time to identify all dubious data.

### 3.3 Identifying Dubious Data

Having obtained the set of single target interval association rules, we utilize these rules to clean the dataset. For each data tuple, we compare the observed target value with the predicted value. If the target value deviates from the predicted value, we say that the target value is highly suspicious.

Note that we allow the use of different interestingness measures, such as confidence and Piatetsky-Shapiro's rule-interest, to resolve such cases that no rule matches the current data tuple. It has been observed that different interestingness measures affect the number of false alarms.

## 4 Experiments

We implemented the techniques presented in this paper in Java. Experiments to evaluate the proposed method are carried out on a Pentium IV 1.6 GHz with 128 RAM running Windows 2000.

### 4.1 Synthetic Data Set

**A. Experiment Setup.** We generated 30,000 tuples with five attributes A, B, C, D, E such that A, E are categorical attributes, while B, C and D are numeric attributes. The values of these attributes are generated based on some pre-defined pattern templates, for example, pattern "C = [0.0 2.0], A = 1 → B = [7.0 8.0]". Note that the unit of B, C, D are set the same. Once the dataset has been generated, we perform 3-fold cross-validation. We split the dataset into two parts: one is the training dataset for building the interval association rules; the other part is reserved for testing purposes. The ratio between the training and testing data set is 2:1. The experiments are repeated three times and the results are averaged.

Our evaluation criterion is the accuracy of detection defined as follows:

$$DetectionAccuracy = \frac{detected\ dubious\ error\ amount}{total\ dubious\ error\ amount}$$

In the experiments, we compare our approach(IAR) to other four approaches, namely classical association rule(CAR), statistical method (SM), clustering [15] (CL) and pattern identification [1] methods(PI). The CAR applies only to categorical attributes. For SM, values outside (mean ± 3 \* standard deviation) are

considered dubious according to the Central Limit Theorem. For CL, we use the techniques in [15], which can cover both numeric and categorical attributes. Finally, for PI, [1] provides a linear approach for deviation detection.

**B. Accuracy.** In this set of experiments, we test the effects of user-defined parameters (diameter threshold, minimum support and confidence, the amount of noise data in testing and training dataset) to the detection accuracy. Note that noise data refer to those data tuple that deviates from the pattern templates. In addition, as the unit of continuous data are the same,  $d_n$  is uniformed. And  $d_c$  is equal to zero in the experiments.

Figure 5 shows that the accuracy of IAR decreases when  $d_n$  (the interval size) increases. From the figure, we note that when  $d_n \geq 1.5$ , SM outperforms IAR. The reason is that if the interval size is enlarged, the rules generated are too general and many noise data can match these rules. As a result, detection accuracy is badly affected. The rest methods do not perform well.

From Figure 6 and 7, we see how the detection accuracies vary with *minsup* and *minconf*. When *minsup*  $\leq 0.05$  or *minconf*  $\leq 0.48$ , it is clear that when *minsup* and *minconf* increase, the accuracy of IAR decreases. After that, the accuracies of IAR begin to decrease. The reason is that too high support or confidence makes rules related to fewer attributes. This leads to the dubious data of some attributes which is uncovered by the rules missing.

Figure 8 shows how varying the amount of noise data in the training set influences the detection accuracy. Interestingly, IAR accuracy is not affected by the amount of of noise data in the training set.

Next we vary the amount of noise in the test set. Figure 9 shows the results. IAR is the winner in all five methods. When the amount of noise increases, the accuracy of SM decreases as there is a large variation in the distribution of the test set. The performance of PI, once again, is not stable. CL, on the other hand, focuses on the relationship of the whole data tuple in the cluster model. Hence, it misses all those deviations in the lower dimensions.

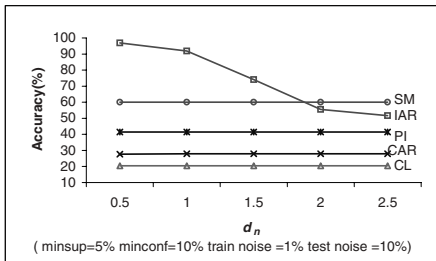


Fig. 2. Varying Interval Size

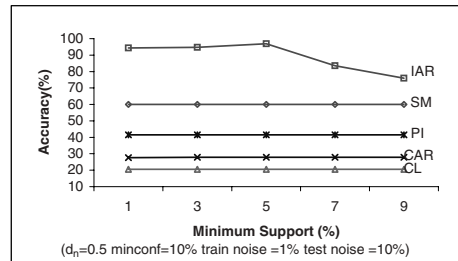


Fig. 3. Varying *minsup*

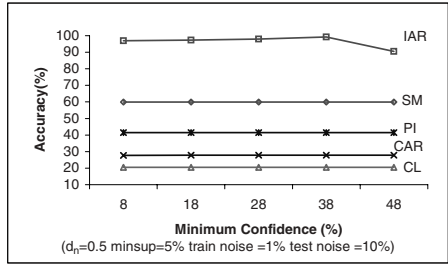


Fig. 4. Varying *minconf*

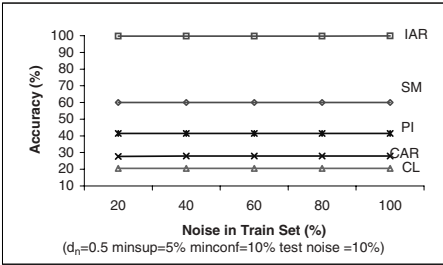


Fig. 5. Varying Noise in Train Set

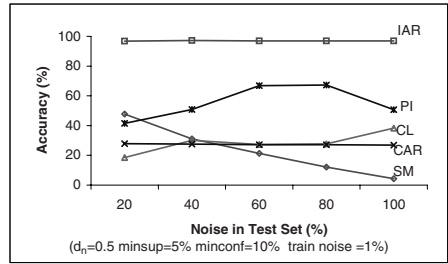


Fig. 6. Varying Noise in Test Set

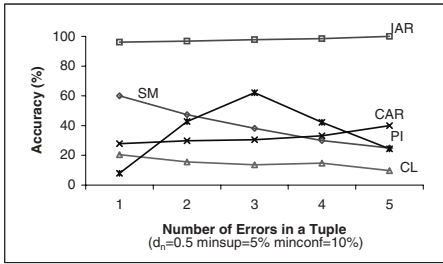


Fig. 7. Varying Errors in a Tuple

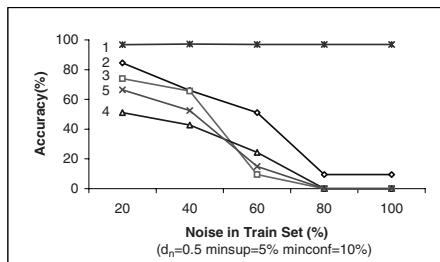
**C. Scalability.** In this subsection, we examine the scalability of the proposed method. Figure 10 shows the results. We see that the increase in the number of errors in a data tuple almost has no influence on the IAR accuracy. Note that the noise in train set and test set are 1% and 10% respectively. On the other hand, the accuracy of CAR improves with the increase in the number of errors per tuple while the accuracies of the other three methods degrade as the number of errors increases.

Then we vary the errors for both total records and a tuple. When the noise in a dataset increases from 20% to 100%, the accuracy almost has no change when there is one error per tuple. On the contrary, the accuracies decrease even reaches zero when the error per tuple  $\geq 2$ . It is attributable to the rules generated largely influenced by such cases which deviate from pattern templates.

## 4.2 Real-Life Data Set

Finally, we run our algorithm on a real-life dataset. This dataset contains the grade information on 759 students. It consists of 5 numeric attributes and 4 categorical attributes (Sex, CAP, SAP, A-Level Point, GP Grade, Subj1 Grade, Subj2 Grade, Subj3 Grade, Subj4 Grade). It is worthwhile to note that this dataset is thought to be free of error.

3024 rules are generated after running our algorithm on the dataset. Using these rules, 30 dubious data values are highlighted. Table 3 shows a sample of



**Fig. 8.** Varying the Number of Errors in a Tuple

**Table 3.** Dubious Data Values in Real World Data Set

Sex=M, CAP=3.16, SAP=3.5, ALevel=61, GP=5, Subj1=6, Subj2=A, Subj3=B, Subj4= <b>X</b> Expected Value: Subj4 = {A, B, C}
Sex=M, CAP=4.25, SAP=4.33, ALevel=55, GP=3, Subj1=5, Subj2= <b>X</b> , Subj3=B, Subj4=D Expected Value: Subj2 = {A, B, C, D, E}
Sex=M, CAP=4.23, SAP=4.33, ALevel=62, GP=6, Subj1= <b>9</b> , Subj2=A, Subj3=B, Subj4=A Expected Value: Subj1 = [1.0, 8.0]
Sex=M, CAP=3.41, SAP=3.7, ALevel=52, GP=5, Subj1=1, Subj2=C, Subj3= <b>O</b> , Subj4=A Expected Value: Subj3 = {A, B, C, D, E }

the dubious data values found. We first list the actual record content, and then show the set of expected values for an attribute. The dubious attribute value is highlighted in bold. The first two examples indicate values for absence from the exam. The third example turns out to be correct. This is acceptable as the actual grade is only one away from the expected range. The fourth example, on the other hand, warrants additional investigation. The result is certainly encouraging as we do not expect to find any dubious data in this dataset.

## 5 Conclusions

In this paper, we have presented a framework to solve dubious data problem. The main idea of the proposed method is to utilize interval association rules to detect dubious data values for both numeric and categorical attributes. Our approach consists of three steps: preprocessing, rule generation, and dubious data detection. In the preprocessing step, we fill in the missing values, standardize different data format and perform data transformation. During the rule generation pro-

cess, we build interval association rules through clustering techniques. Finally, we perform dubious data detection by applying the interval association rules. Experiment results indicate that our approach is both effective and efficient in detecting dubious data.

## References

1. A.Arning, R.Agrawal, and P.Raghavan. A linear method for deviation detection in large databases. *AAAI*, 1996.
2. A.E.Monge. Matching algorithms within a duplicate detection system. *IEEE Bulletin on Data Engineering*, 2000.
3. A.Mauricio, Hernández, and S.J.Stolfo. The merge/purge problem for large databases. *SIGMOD*, 1995.
4. C.C.Aggarwal and P.S.Yu. Outlier detection for high dimensional data. *SIGMOD*, 2001.
5. E.Knorrr and R.Ng. Algorithms for mining distance-based outliers in large data sets. *VLDB*, 1998.
6. E.Rahm and H.H.Do. Data cleaning: Problems and current approaches. *IEEE Bulletin on Data Engineering*, 2000.
7. J.Maletic and A.Marcus. Automated identification of errors in data sets. 2000.
8. J.Maletic and A.Marcus. Utilizing association rules for the identification of errors in data. 2000.
9. M.M.Breuing, H.-P.Kriegel, R.T.ng, , and J.Sander. Lof: Identifying density-based local outliers. *SIGMOD*, 2000.
10. R.Agrawal, T.Unuekubsum, and A.Swami. Mining association rules between sets of items in large databases. *VLDB*, 1993.
11. R.J.Miller and Y.Yang. Association rules over interval data. *SIGMOD*, 1997.
12. R.Kimball. Dealing with dirty data. *DBMS Online*.
13. R.Srikant and R.Agrawal. Mining quantitative association rules. *VLDB*, 1995.
14. S.Ramaswamy, R.Rastogi, , and K.Shim. Efficient algorithms for mining outliers from large data sets. *SIGMOD*, 2000.
15. T.Chiu, D.P. Fang, J. Chen, and Y. Wang. A robust and scalable clustering algorithm for mixed type attributes in large database environment. *SIGKDD*, 2001.
16. V.Barnett and T.Lewis. Outliers in statistical data. 1994.

# Mining Web Sequential Patterns Incrementally with Revised PLWAP Tree

Christie I. Ezeife\* and Min Chen

School of Computer Science, University of Windsor,  
Windsor, Ontario, Canada N9B 3P4  
cezeife@uwindsor.ca

**Abstract.** Since point and click at web pages generate continuous data stream, which flow into web log data, old patterns may be stale and need to be updated. Algorithms for mining web sequential patterns from scratch include WAP, PLWAP and apriori-based GSP. An incremental technique for updating already mined patterns when database changes, which is based on an efficient sequential mining technique like the PLWAP is needed.

This paper proposes an algorithm, Re-PL4UP, which uses the PLWAP tree structure to incrementally update web sequential patterns. Re-PL4UP scans only the new changes to the database, revises the old PLWAP tree to accommodate previous small items that have become large and previous large items that have become small in the updated database without the need to scan the old database. The approach leads to improved performance.

**Keywords:** Incremental Mining, sequential mining, frequent patterns, PLWAP tree, Scalability

## 1 Introduction

When data are inserted or deleted from a database (like access patterns in web access log), previous patterns may no longer be interesting and new interesting rules could appear in the updated database. Incremental mining of sequential patterns is the process of generating new patterns in the updated database (old + new data) by using only the updated part (new data) and previously generated old patterns.

Sequential mining, unlike general association rule mining pays attention to the order the items occur as well [2]. For example, given a small web access log that recorded user accesses to 8 sites represented as sites  $\{a, b, c, d, e, f, g, h\}$  as shown in Table 1.

Each transaction consists of items (or attributes, e.g., web sites visited), the association rule rule  $a \rightarrow b$  asserts that if site  $a$  is visited, then, site  $b$  is visited.

---

\* This research was supported by the Natural Science and Engineering Research Council (NSERC) of Canada under an Operating grant (OGP-0194134) and a University of Windsor grant.



**Table 1.** The Example Database Transaction Table with Frequent Sequences

TID	Web access Seq.	Frequent subseq with $s = 50\%$
100	abdac	abac
200	aebcace	abcac
300	baba	baba
400	afbacfc	abacc
500	abegfh	ab

Each of these 8 sites represents an item in the database. An itemset  $X$  is called an  $i$ -itemset if it contains  $i$  items. The support of a rule is defined as the percentage of transactions (records) that contain the sets  $X$  and  $Y$ , while its confidence is the percentage of all database transactions containing “ $X$ ”, that also contain “ $Y$ ”. All items with support higher than a specified minimum support are called large or frequent itemsets. While minimum support is used to mine frequent patterns, only rules with confidence higher than the minimum confidence are retained. An example of a sequential pattern from the database of Table 1 is “if site  $a$  is accessed in a record, it is often followed by an access to site  $b$ ”. While the order is important in sequential pattern, items in a sequence do not necessarily need to be consecutive and an item can be repeated in one sequence. A sequence also has subsequences, e.g.,  $abacc$ ,  $afb$  are some subsequences of the sequence  $afbacfc$ . Frequent sequences are those with support equal or higher than minimum support (minsupport).

When data are inserted or deleted from a database, previous patterns may no longer be interesting and new interesting rules could appear in the updated database. Incremental mining of sequential patterns is the process of generating new patterns in the updated database (old + new data) using only the updated part (new data) and previously generated patterns.

### 1.1 Related Work

Few existing algorithms [1, 4, 6] for incremental sequential patterns mining are apriori-based rather than tree-based. They are composed of iteratively: (1) generating all large itemsets in the database and (2) generating sequential patterns in the database according to the large itemsets generated in the first step. ISE algorithm is proposed by [4] for incremental sequential patterns mining and scans the database several times. ISM [6] is also an apriori-like algorithm. It still needs to rescan the entire updated database many times if previous small items become large after database update. When the 1-sequence becomes large, the updated data becomes explosive. WAP tree algorithm [5] scans the database only twice to build the WAP tree without generating candidate sets. It then, mines the WAP tree to extract sequential patterns. PLWAP algorithm uses a preorder linked version of WAP tree and an algorithm to eliminate the need to recursively re-construct intermediate WAP trees during mining as done by WAP tree technique. Neither WAP nor PLWAP algorithm is used for incremental mining.

## 1.2 Contributions

This paper proposes an algorithm named Re-PL4UP (Revised PLWAP FOR UPdated sequential mining). This algorithm applies the PLWAP-tree [3] to the incremental sequential mining problem. The Re-PL4UP algorithm eliminates the need to re-scan the old database when new changes arrive, in order to update old patterns. The Re-PL4UP algorithm uses the information from original PLWAP tree, without the need to re-scan the entire database (old + new data) in order to update old patterns. The approach is to initially build a PLWAP tree that is based on minimum support,  $s$  and which remembers the position codes of small items. Revising the PLWAP tree will entail traversing the tree to insert new nodes, delete nodes that are no longer frequent, changing links and re-building the prefix linkages used during mining. A mirror copy of all modified and newly inserted branches of the tree is mined and the patterns from this mirror copy are used to update the previously mined patterns. This yields better performance and allows for application scalability.

Section 2 first presents the proposed incremental sequential mining algorithm, Re-PL4UP. Section 3 discusses an example mining of a database and its update using the proposed Re-PL4UP algorithm, section 4 presents performance analysis of the algorithm, and finally, section 5 presents conclusions and future work.

## 2 The Proposed Incremental Re-PL4UP Algorithm

The algorithm, Re-PL4UP being proposed for mining frequent sequential patterns incrementally uses PLWAP tree structure. Section 2.1 discusses the algorithm Re-PL4UP. Let  $F$  and  $S$  represent previous large (frequent) items and previous small items in original database respectively;  $F'$  and  $S'$  represent updated large (frequent) items and updated small items in the updated database  $U$ , each event (item) in updated database  $U$ , belongs to one of these six categories of items: (1). Frequent in old database,  $DB$  and are still frequent in updated database (old + new data),  $U$ , ( $F \rightarrow F'$ ); (2). Frequent in old  $DB$  but small in updated database, ( $F \rightarrow S'$ ); (3). Small in old  $DB$  but frequent in updated database, ( $S \rightarrow F'$ ); (4). Small in old  $DB$  and still small in updated database, ( $S \rightarrow S'$ ); (5). New and frequent in updated database ( $\emptyset \rightarrow F'$ ); (6). New and small in updated database ( $\emptyset \rightarrow S'$ ).

### 2.1 Mining Incremental Patterns with Re-PL4UP

This algorithm scans only the changes to the database (db). Next, it uses frequent items in the changes to the database to update the old PLWAP tree of the original database before mining. The most important update made to the old PLWAP tree are for two classes of items namely: items in category 2, which are,  $F \rightarrow S'$ , that now need to be deleted from the old tree; and the items in category 3, which

are  $S \rightarrow F'$  that need to be inserted into the tree. The Re-PL4UP approach is to take advantage of the position codes property of the PLWAP tree and during initial construction of the PLWAP tree, store the list of position codes of all small items. During update, these unique position codes are used to re-insert the previous small items in proper positions in the tree without re-scanning the old database. The updated patterns are now the union of the old patterns, the patterns from modified branches of old Re-PL4UP tree and the patterns from the new incremental database tree.

## 2.2 Mining Initial Frequent Patterns with PL4UP

This section presents the sequence of steps for mining initial frequent patterns using the proposed Re-PL4UP algorithm, as well as how to mine it incrementally, while an example application of the algorithm is given in section 3. Given a web access sequence database (WASD) or its equivalent, a minimum support,  $s$ , the proposed Re-PL4UP algorithm will take the following steps in mining initial frequent patterns.+

1. Construct initial PLWAP tree using minimum support,  $s$  obtaining frequent 1-items (with support  $\geq s$ ),  $F_1$ , the frequent 1-items, the Small 1-items (with support  $< s$ ),  $S_1$ . Then, during construction of the PLWAP tree, we shall define a small item code profile for every small 1-item in the  $S_1$  list by including a list of position codes of the PLWAP tree indicating the position that this small item would have been on, in the tree if it were frequent.
2. We construct a PLWAP tree [3] using the  $seq_s$  from first step above and the frequent  $F_1$  items for header link connections and define the small item code profile for small items  $S - code^{DB}$ . This is called  $Re - PL4UP^{DB}$  tree.
3. We now mine the  $Re - PL4UP^{DB}$  tree for frequent patterns (called  $FP^{DB}$ ), by extracting all patterns with support greater than or equal to  $s$ .

## 2.3 Steps in Incremental Mining of Frequent Patterns with Re-PL4UP

When new transactions are inserted into or deleted from the database, the steps for updating old patterns are given below. The input data for this process are: (1) the changed database,  $db$ , (2) the old database DB tree,  $Re - PL4UP^{DB}$ , (3) the candidate 1-items,  $C_1^{DB}$ , (4) minimum supports,  $s$ , (5) the frequent 1-items,  $F_1$ , (6) the small 1-items,  $S_1$ , (7) the frequent patterns,  $FP^{DB}$ . The steps in incrementally mining the database using mostly changed  $db$  and old available patterns are presented in Figure 1 while an example mining with this algorithm is presented in section 3.

## 3 Mining an Example Database with Re-PL4UP Algorithm

Suppose we have a database DB with set of items,  $I = \{a, b, c, d, e, f, g, h\}$  and minimum support = 50% of DB transactions. A simple database transaction

**Algorithm 21** (*Re-PL4UP-Mines Web Log Sequences Incrementally*)**Algorithm Re-PL4UP()**

**Input:** original database, DB, Incremental database, db, minimum support  $\lambda$  ( $0 < \lambda \leq 1$ ) original DB tree  $Re - PL4UP^{DB}$ , old frequent pattern, FP, old candidate lists  $(C_1, F_1, S_1)$ , small code profile  $S - code^{DB}$ .

**Output:** updated frequent patterns for database, U (FP'),  $Re - PL4UP^{DB}$  tree.  
updated candidate lists  $(C'_1, F'_1, S'_1)$ . updated small code profile  $S - code^U$ .

**Intermediate data:** incremental db candidate lists  $(C_1^{db}, F_1^{db}, S_1^{db})$

**begin**

- (1) Update all candidate lists as follows:  
 $C'_1 = C_1 \cup C_1^{db}$ ;  $s' = \lambda$  of  $|DB| + |db|$ .  
 $F'_1$  = elements in  $C'_1$  with support  $\geq s'$   
 $S'_1$  = elements in  $C'_1$  with support  $< s'$ .  
 $F_1^{db} = C_1^{db} \cap F'_1$ ;  $S_1^{db} = C_1^{db} \cap S'_1$ .
- (2) Classify items in the updated data, U into one of 6 classes as:  
 $F_1 \cap F'_1$  are in class  $F \rightarrow F'$ ;  $F_1 \cap S'_1$  are in class  $F \rightarrow S'$ .  
 $S_1 \cap F'_1$  are in class  $S \rightarrow F'$ ;  $S_1 \cap S'_1$  are in class  $S \rightarrow S'$ .  
 $F'_1 - F_1$  are in class  $\emptyset \rightarrow F'$ ;  $S'_1 - S_1$  are in class  $\emptyset \rightarrow S'$ .
- (3) Modify the old  $Re - PL4UP^{DB}$  tree such that all  $F \rightarrow S'$  items are deleted from the tree, and all  $S \rightarrow F'$  are inserted into tree using the  $S - code^{DB}$ .
- (4) Mine modified branches of  $Re - PL4UP^{DB}$  to get frequent patterns  $Re - FP^{DB}$ .
- (5) Construct and mine small  $Re - PL4UP^{db}$  to get frequent patterns  $Re - PL4UP^{db}$ .
- (6) Combine the three frequent patterns to obtain FP' as:  
 $FP' = FP^{DB} \cup Re - FP^{DB} \cup FP^{db}$
- (7) Insert the frequent sequence transactions from the incremental database into the original  $Re - PL4UP^{DB}$  tree to update it; update the links and small code profiles.

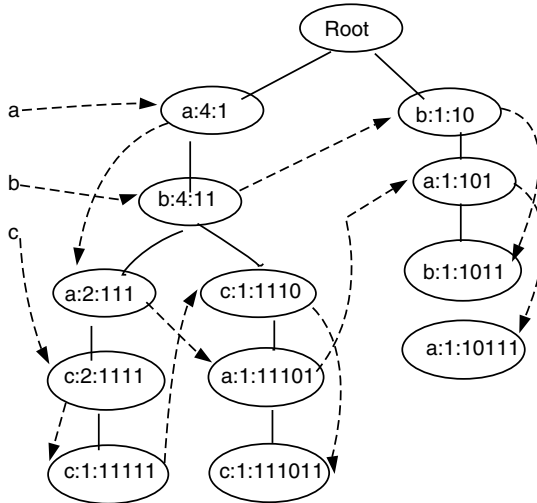
end // of Re-PL4UP //

**Fig. 1.** The Re-PL4UP Algorithm

table for illustrating the idea is given in the first two columns of Table 1. The first process is to build the initial  $Re - PL4UP^{DB}$  tree [3] using sequences in Table 1 with support greater than or equal to the tolerance support  $s$  of 50% (equivalent to 3 transactions in Table 1). The  $Re - PL4UP^{DB}$  tree is built the same way the PLWAP tree is built. To build, first scan the database sequence (column 2 of the Table) once to obtain the candidate 1-items, 1-frequent and 1-small lists with their supports as:  $C_1 = \{a:5, b:5, c:3, d:1, e:2, f:2, g:1, h:1\}$ .  $F_1 = \{a, b, c\}$ .  $S_1 = \{e, f, d, g, h\}$ . Then, scan the web access database (column 2 of Table), a second time to create a frequent sequence from each transaction, the frequent sequence ( $seq_s$ ) that includes all items with support greater or equal to support,  $s$  ( $seq_s$  is shown on column 3 of Table 1). insert each of the  $seq_s$  transactions in the tree with their count and position code to obtain the PLWAP tree in (Figure 2). Each node is described as (a:1:1) standing for (label of the

node:count of the node:position code of the node). In defining the position code of a node, the PLWAP algorithm applies the rule that the root of the tree has a null position code, but every other node has a position code that is equivalent to appending '1' to the code for this node's parent if this node is the leftmost child of its parent, otherwise, its position code is obtained by appending '0' to the position code of its nearest left sibling. While inserting the frequent items in the sequence, the algorithm checks the original transaction to mark location of small items in the transaction. For example, small item  $d$  in the original first transaction  $abdac$  would have had the position code (d:1:111) in the created branch if this  $d$  were frequent. It will write this position code in the small item code profile for item  $d$  as  $S-code^d = \{111\}$ . The position code of a node does not change unless the node moves. The complete small item code profiles for all small items after inserting all frequent sequences in the tree are:

$S-code^d = \{11\}$  or  $\{3\}$ ,  $S-code^e = \{110, 1101111, 1110\}$  or  $\{6, 223, 14\}$ ,  $S-code^f = \{1100, 11001111, 111011\}$  or  $\{12, 207, 59\}$ ,  $S-code^g = \{11101\}$  or  $\{29\}$ ,  $S-code^h = \{1110111\}$  or  $\{119\}$ . Note that these position codes can also be recorded in their decimal number equivalents as shown above. After building the tree, a pre-order traversal mechanism (visit root, visit left subtree, visit right subtree) is used to add a pre-order linkage on the tree for all frequent 1-items,  $F_1 = \{a, b, c\}$ . The broken lines on Figure 2 starting with each frequent  $F_1$  item is used to show the pre-order linkage between nodes of this type. In step 3, the  $Re-PLAUP^{DB}$  begins the mining process. It starts following the header linkage of the first frequent item, "a", and obtains the support of this prefix subsequence "a" as the sum of the counts of all first "a" nodes in the current a:suffix root set, which are the tree branches rooted at (a:4:1 and a:1:101). If the sum of the counts of these first  $a$  nodes on different branches of the tree at the level



**Fig. 2.** The Re-PL4UP tree with Support  $s$  for the Example Database

of the tree under consideration is greater or equal to support of 3, we include this sequence in the  $FP^{DB}$  frequent pattern list. The “a” is frequent because this suffix root set has a total count of 5. Next, we shall continue to check down the suffix trees under use to see if the sequences (aa, ab, ac) are also frequent. The a-header link serves the purpose of tracking and constructing these root sets during mining. Process continues in a similar fashion recursively checking for all patterns beginning with “a”, “b” and “c”. After checking all patterns, the list of  $FP^{DB}$  mined based on the support of 3 is:  $FP^{DB} = \{a:5, aa:4, aac: 3, ab:4, ac:3, aba:4, abac:3, abc:3, b:5, ba:4, bac:3, bc:3, c:3\}$ .

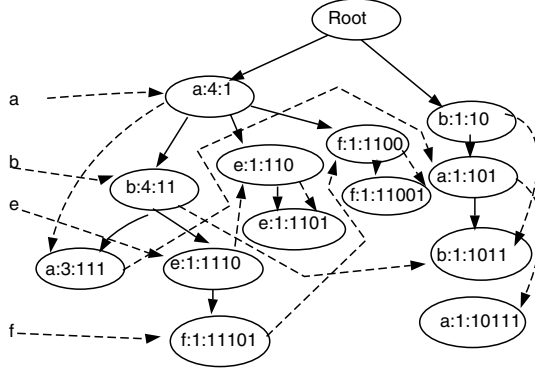
### 3.1 Mining Updated Database with Re-PL4UP Tree Algorithm

Assume that the original database, DB of Table 1 is updated with the records in the changes to database table shown as Table 2, the objective of the *Re-PL4UP<sup>DB</sup>* algorithm is to use old rules,  $FP^{DB}$  from the previous section, with the new changes to database and other intermediate information from the previous section like candidate 1-items, frequent 1-items, and small 1-items, small item code profiles  $S-code^{DB}$ , to compute the new frequent patterns in the entire updated database, using the same support, s, of 50%. Thus, the *Re-PL4UP<sup>DB</sup>* algorithm mines the updated database incrementally as:

**Table 2.** The Changes to Database Transaction Table

TID	Web access Seq.	Frequent subseq with s = 50%	Frequent subseq with t = 0.6s
700	bahefg	bahefg	bahefg
800	aegfh	aegfh	aegfh

1. Update all intermediate candidate lists and patterns as follows:  $C'_1 = C_1 \cup C_1^{db}$ .  $C_1 = \{a:5, b:5, c:3, d:1, e:2, f:2, g:1, h:1\}$ .  $C_1^{db} = \{a:2, b:1, e:2, f:2, g:2, h:2\}$ , thus,  $C'_1 = \{a : 7, b : 6, c : 3, d : 1, e : 4, f : 4, g : 3, h : 3\}$ .  $F'_1 = \{a : 7, b : 6, e : 4, f : 4\}$  and  $F_1^{db} = C_1^{db} \cap F'_1 = \{a:2, b:1, e:2, f:2\}$ .  $S'_1 = \{c : 3, d : 1, g : 3, h : 3\}$ .  $S_1^{db} = C_1^{db} \cap S'_1 = \{g:2\}$ .
2. Classify items in the updated database U, into one of the defined six categories as:(i)  $F_1 \cap F'_1 = \{a, b\}$ , (ii)  $F_1 \cap S'_1 = \{c\}$  (iii)  $S_1 \cap F'_1 = \{e, f\}$ , (iv)  $S_1 \cap S'_1 = \{d, g, h\}$ , (v)  $F'_1 - F_1 = \emptyset$ , and (vi)  $S'_1 - S_1 = \emptyset$ .
3. Modify the old *Re-PL4UP<sup>DB</sup>* tree to delete from items in category  $F \rightarrow S'$ , constituting  $\{c\}$ , to insert all items in the category  $S \rightarrow F' = \{e, f\}$  using their small code profile. Every small item code profile has a unique position in the current tree, determined by any matching prefix of its binary position code. Thus, if the position profile code is 111011 and we can find a node position in the current tree for the prefix 1110, then, we insert the small-to-large item there. The new code profile of the item is the physical code in the tree. The small code profile list is updated such that the updated codes for the small-to-large items  $\{e, f\}$  after tree modification are:  $S-code^e = \{110,$



**Fig. 3.** The Revised PL4UP tree After Deletion and Insertion of Items with Changed Status

1101, 1110} and  $S - codef' = \{1110, 11001, 11101\}$ . Finally, the frequent header linkages of the frequent items in the modified tree are re-constructed. The revised tree is shown as Figure 3.

4. Mine only the modified branches of the revised tree,  $Re - PL4UP^{DB}$  (which is the left branch of Root in Figure 3) to obtain the frequent sequences called  $Re - FP^{DB}$  as  $\{aee:1, abef:1, aff:1, ae:2, af:2, be:1, bf:1, bef:1, ef:1, ee:1, ff:1, e:2, f:2\}$ .
5. Construct the small  $Re - PL4UP^{db}$  tree using only the changes to the database, db. Mine this  $Re - PL4UP^{db}$  to obtain the  $FP^{db}$  patterns. The tree for small db is based on  $F_1^{db} = C_1^{db} \cap F'_1 = \{a, b, e, f\}$ . Following this process, the useful mined  $FP^{db} = \{b:1, a:2, h:2, e:2, f:2, g:2, ba:1, be:1, bf:1, aef:2, bef:1, ae:2, af:2, ef:2, bf:1\}$ .
6. Combine the frequent patterns from (1) old database, DB, and (2) modified branches of old DB and (3) changes to the database, db, keeping only those patterns that have support  $\geq s'$  (which is 4) in updated database.  $FP'$  are patterns in  $FP^{DB} \cup Re - FB^{DB} \cup FP^{db}$  with support greater or equal to  $s'$  (of 4 transactions). Thus,  $FP' = \{a:7, aa:4, ab:5, aba:4, b:6, ba:5, ae:4, af:4, e:4, f:4\}$ .
7. Finally, the frequent sequences in the incremental database, db, which are (baef, aef) are inserted into the main revised  $Re - PL4UP^{DB}$  to keep it up-to-date, while the small code profile of the new small items in the new changes as well as the small code profiles of all deleted nodes from the tree are recorded in the small code profiles.

## 4 Experimental and Performance Analysis

A performance comparison of Re-PL4UP, PLWAP and ISE algorithms was conducted. All these three algorithms were implemented and run on the same datasets generated using the resource code for generating synthetic datasets

downloaded from <http://www.almaden.ibm.com/cs/quest/syndata.html>. The experiments were conducted on a Pentium 4 PC machine with 256 megabytes of main memory running Windows operating system. The programs were written in C++ under visual C++ environment. The number of transactions (D) in this dataset is sixty thousand records, that is  $|D| = 60,000$  records, the average size of transactions (length of sequences) (T) is 10,  $|T| = 10$ , average length of maximal pattern (that is, average number of items in the longest frequent sequence) (S) is 6, or  $|S| = 6$ , number of items or events (N) (the total number of attributes) is one thousand,  $N=2000$ . Assume the size of updated (inserted) dataset is 20,000 records, and the support thresholds are varied between 1% and 20%. An experimental result is shown in Table 3.

**Table 3.** Execution Times for Dataset at Different Supports

Algorithms	CPU Time (in secs) at Supports of				
	1	2	3	4	5
ISE	4000	1405	515	188	101
PLWAP	212	112	80	66	60
Re-PL4UP	164	69	40	27	14

### Experiment 2: Execution Time for Databases with Different Sizes

We use different database sizes that vary from 20k to 100k to compare the three algorithms. The minimum support 20% is used for Re-PL4UP, ISE and PLWAP and the result of the experiment is shown in Table 4. The five datasets used for the experiment are T10.S5.N2000.D20K, T10.S5.N2000.D40K, T10.S5.N2000.D60K, T10.S5.N2000.D80K, and T10.S5.N2000.D100K.

**Table 4.** Execution Times at Different Transaction Sizes on Support 16%

Algorithms (times in secs)	Different Changed Transaction Size				
	20K	40K	60K	80K	100K
ISE	100	189	285	378	470
PLWAP	29	51	81	98	128
Re-PL4UP	19	33	54	83	105

## 5 Conclusions and Future Work

Re-PL4UP algorithm proposed in this paper, modifies an existing PLWAP tree by utilizing the metadata of old database transactions as well as old mined frequent patterns in order to incrementally update web log sequential patterns. One major contribution of work is the technique for efficiently using position codes of small items in database sequences to restore information about previous small items that were not stored in the tree, when the database is updated



and these items become frequent, without re-scanning old database. Experiments show that Re-PL4UP performs better than existing incremental sequential mining algorithms and no huge candidate itemsets need to be generated. like the old patterns and tree for mining the updated database. Future work should investigate applying technique to distributed and parallel mining that may involve continuous time series data, and to web content and text mining.

## References

1. Agrawal, R. and Srikant, R.: Mining Sequential Patterns, Proceedings of the 11th Int'l Conference on Data Engineering, Taipei, Taiwan, March 1995, pp. 3-14.
2. Han, J., Kamber, M.: Data Mining: Concepts and Techniques Morgan Kaufmann, 2001.
3. Lu, Yi., Ezeife, C.I.: Position Coded Pre-Order Linked WAP-Tree for Web Log Sequential Pattern Mining, Proceedings of The 7th Pacific- Asia Conference on Knowledge Discovery and Data Mining (PAKDD 2003), Seoul, Korea, Apr. 30-May 2 2003, , pp. 337-349.
4. Masegla, F., Poncelet, P., Teisseire, M.: Web Usage Mining: How to Efficiently Manage New transactions and New Customers. Rapport de Recherche LIRMM, 18 pages, Fevrier 2000. Version courte dans Proceedings of the 4th European Conference on Principles of Data Mining and Knowledge Discovery (PKDD'00), Lyon, France, September 2000, pp.530-535.
5. Pei, Jian., Han, Jiawei., Mortazavi-asl, Behzad., Zhu, Hua.: Mining Access Patterns Efficiently from Web Logs. Proceedings 2000 Pacific-Asia Conf. On Knowledge Discovery and Data Mining (PAKDD'00), Kyoto, Japan, April 2000.
6. Parthasarathy, S., Zaki, M.J., Ogihara, M., Dwarkadas, S. Incremental and Interactive Sequence Mining, In Proc.(1999) of the 8th International Conference on Information and Knowledge Management (CIKM99), 251- 258, Kansas City, MO, November 1999, pp.530-535.

# Mining Frequent Items in Spatio-temporal Databases\*

Cheqing Jin<sup>1</sup>, Fang Xiong<sup>1</sup>, Joshua Zhexue Huang<sup>2</sup>,  
Jeffrey Xu Yu<sup>3</sup>, and Aoying Zhou<sup>1</sup>

<sup>1</sup> Department of Computer Science and Engineering, Fudan University, P.R.C  
{cqjin,xfang,ayzhou}@fudan.edu.cn

<sup>2</sup> E-Business Technology Institute, The University of Hong Kong  
jhuang@eti.hku.hk

<sup>3</sup> Department of S.E.E.M., The Chinese University of Hong Kong  
yu@se.cuhk.edu.hk

**Abstract.** It is important to retrieve aggregate information in spatio-temporal applications. Recently, some applications, such as decision support systems, also require to mine frequent items based on a dataset within a query region during a query interval. Because of unbounded space requirement and slow response time, executing query based on operational databases becomes inapplicable.

In this paper, we define the problem formally and give out a novel solution to overcome the above two disadvantages. Recently, some algorithms are proposed to mine frequent items from a summarization(sketch) of a mass dataset. In our solution, one of latest sketches is integrated with a spatio-temporal index to provide good performance.

## 1 Introduction

Spatio-temporal databases play an important role in applications that involve space and time. Road traffic supervision and location based mobile services in large cities are typical examples. Most reported works have been focused on the problems of efficiently querying aggregate information about objects qualified on spatio-temporal conditions, such as the *spatio-temporal count* problem and *spatio-temporal sum* problem [16].

Mining frequent occurrence items from a large data set has been a popular research issue in databases, data mining and computer networks [1][7][8]. It is also a critical research problem in spatio-temporal applications. For example, a retail chain company such as 7-Eleven has many shops scattered in big cities. Individual shops at different locations result in different sales at different time periods. Queries about the aggregate sales of different product categories at shops in different regions at different time periods require the transactions being qualified on both spatial and time conditions. For example, the general manager

---

\* This work is supported by the National Hi-Tech Research and Development Program of China under Grant No. 2002AA413310.

uses a mouse to draw a region of the downtown area on the city map and wants to find out the 10 best sold products in the shops in this area in the morning. If the central data warehouse for transactions is very large, the quick response to the dynamic space and time windows is a big challenge in implementing such queries.

To our best knowledge, there is no efficient solutions available till now. In this paper, we propose a new method by using **hCount** sketch (will be introduced in Sect 2.2) to present spatio-temporal cells and a novel structure, sRB-tree, to index such sketches. Our experiment results have shown that this new method could produce good approximation results of spatio-temporal queries and in the meantime significantly reduce the size of datasets.

The rest of the paper is organized as follows. In Sect 2, we formally define the problem and introduce related work. Our solution is proposed in Sect 3. In Sect 4, we present extensive experimental studies and report our findings. We conclude our work in Sect 5.

## 2 Preliminaries

### 2.1 Problem Definition

Assume that there are a set of  $R$  2D points  $p_1, p_2, \dots, p_R$ , and any two points can not overlap. Every timestamp, multiple transactions are produced in points. Without loss of generality, we assume that all items are integers in a range of  $[1..M]$ . Let  $n_{k,p,t}$  denote the net occurrence of item  $k$  at point  $p$  at timestamp  $t$ . Each transaction  $tran(k, p, t)$  increases counter  $n_{k,p,t}$ , i.e,  $n_{k,p,t} = n_{k,p,t} + 1$ . Let  $N_{p,t}$  denote the sum of net occurrence of all items at point  $p$  at timestamp  $t$ , i.e,  $N_{p,t} = \sum_{i=1}^M n_{i,p,t}$ .

Problem *Spatio-Temporal Frequent-Item query* (abbr. *FI-query*) accepts five parameters, a rectangle  $qr$ , a time interval  $qt$ , a support parameter  $s \in (0, 1)$ , an error parameter  $\epsilon$  such that  $\epsilon \ll s$  and a probability parameter  $\rho$  such that  $\rho$  is near 1. Let  $n_k(q)$  and  $N(q)$  denote the occurrence of item  $k$  and the occurrence of all items at points in  $qr$  during  $qt$  respectively, i.e,  $n_k(q) = \sum n_{k,p,t}$ ,  $N(q) = \sum N_{p,t}$ , where  $p \in qr$  and  $t \in qt$ . The goal of the query is to output items with frequencies  $f_k(q)$  over  $s$ , where  $f_k(q) = n_k(q)/N(q)$ . Moreover, the error must be guaranteed within  $\epsilon$  with probability  $\rho$ .

### 2.2 Related Work

Mining frequent items from a mass itemset has been widely studied and lots of work [4][5][6][12][14] has been completed these years. Some algorithms can solve this problem by maintaining a small sketch (in other words, a hash table), from which frequent items can be outputted anytime, such as **hCount** algorithm. Lemma 1 [12] shows that the precision of **hCount** algorithm can be guaranteed given a small sketch.

**Lemma 1.** [12] In **hCount** algorithm, given  $\frac{\epsilon}{\rho} \cdot \ln(-\frac{M}{\ln \rho})$  counters, each item can be estimated with error no more than  $\epsilon N$  with probability  $\rho$ . Notice that  $N$  is the size of itemset and  $M$  is the distinct number of items.

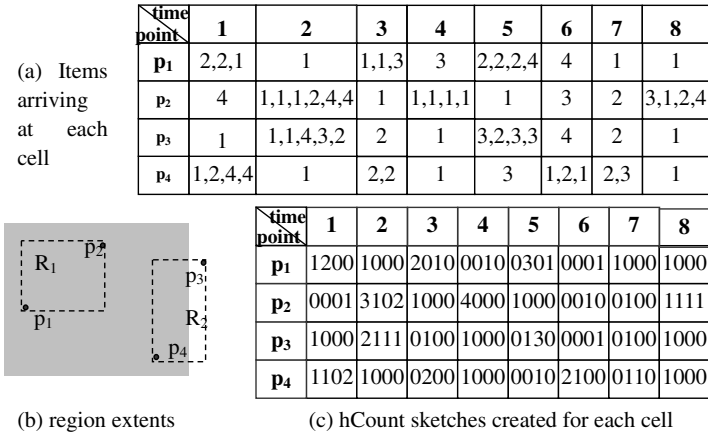
Numerous indexes have been proposed for indexing spatial databases and temporal databases[11][3][13].

Papadias et al.[15] proposed aRB-tree, which is the first access method for efficient spatio-temporal aggregation. Authors organized spatial data by a R-tree. For each entry of R-tree, a B-tree is associated to present temporal data in that region. Some aggregate information is saved in each entry of B-tree.

In [16], authors proposed an approach to query distinct number of objects in spatio-temporal applications by replacing a FM-sketch[9] in each entry of aRB-tree instead of simple aggregation data.

### 3 Our Solution

We will first introduce the global system architecture of our algorithm in Sect 3.1. And then, in Sect 3.2, four strategies are proposed to organize sketches in various ways. Some discussion is given in Sect 3.3.



**Fig. 1.** An small example

#### 3.1 System Architecture

The big idea of this paper is to present the dataset of each cell by **hCount** sketch, and to build an index to facilitate query. Fig. 2 presents our algorithm architecture. One sketch producer (denoted as  $sp_i$ ) is dispatched for a point  $p_i$ . The main task of  $sp_i$  is to present all items arriving in cell  $(p_i, t)$  by a **hCount** sketch  $s_{p_i, t}$ . All sketch producers use same hash functions and create sketches with same size. At the end of  $t$ ,  $s_{p_i, t}$  is transmitted to *insert* processor, where it can be reserved in *database* as a sRB-tree(for details, see Fig. 3), based on which an arbitrary FI-query can be answered.

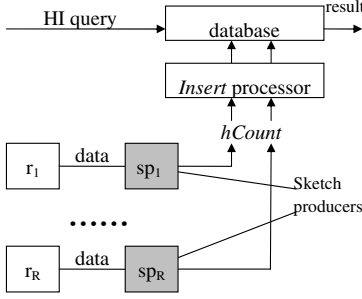


Fig. 2. Architecture

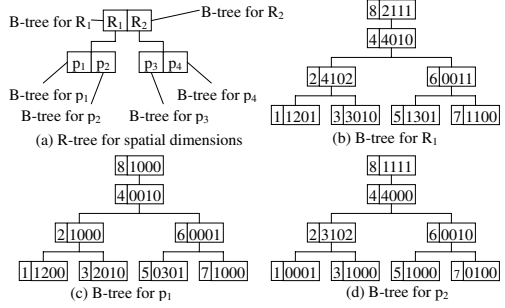


Fig. 3. A sRB-tree example

**Lemma 2.** *Given  $c$  sketches, each presents one of  $c$  cells  $(p_{x1}, t_1), (p_{x2}, t_2), \dots, (p_{xc}, t_c)$ , we can mine frequent items from all items arriving in cells listed above.*

The correctness of this lemma is due to the fact that, the sketch (from the **hCount** algorithm) for the union of several datasets, is identical to the sum of the sketch for each individual dataset. ■

**Lemma 3.** *Given 2 sketches, each presents a set of cells  $set_1, set_2$ . If  $set_1 \supset set_2$ , we can mine frequent items from all items arriving in  $set_1 - set_2$ .*

The correctness of this lemma is due to the fact that **hCount** algorithm owns ability to mine frequent items over updatable stream. ■

**sRB-tree.** sRB-tree(sketch RB-tree) is used to index sketches in database. We will illustrate sRB-tree with a small example. Assume that there are four points ( $p_1, p_2, p_3$  and  $p_4$ ) placed in the example, as shown in Fig. 1(b). And Fig. 1(a) shows items arriving at each point from timestamp 1 to 8. Remember that in the first step, each *sketch producer* must create a sketch every timestamp. For simplicity, each sketch only contains 4 counters, and a item  $i$  is mapped to  $i$ th position of the sketch. Fig. 1(c) shows all sketches created for data in Fig. 1(a). Fig. 3 shows an sRB-tree for this example. The core part of an sRB-tree is a R-tree to index spatio points, just as shown in Fig. 3(a). Notice that  $R_1$  and  $R_2$  are MBR(minimum bounding rectangle) for points  $p_1, p_2$  and  $p_3, p_4$  respectively. Each entry of R-tree is affiliated with a B-trees to record historical sketches. Fig. 3(b)(c)(d) shows three B-trees built for  $R_1, p_1$  and  $p_2$  (B-trees for  $R_2, p_3, p_4$  are omitted due to lack of enough space). Sketches in leaf entry of R-tree present items created in each point, while sketch in non-leaf entry of R-tree is the sum of corresponding sketches in child nodes. For example, the root node of  $R_1$  is associated with sketch (2111), which is the sum of the (1000)(at the root of  $p_1$ ) and (1111)(at the root of  $p_2$ ).

In Sect 3.2, we provide various strategies to organize sketches in B-tree other than Fig. 3.

**Query Framework.** According to Lemma 2,3, it is easy to answer FI-query from an sRB-tree by following steps.

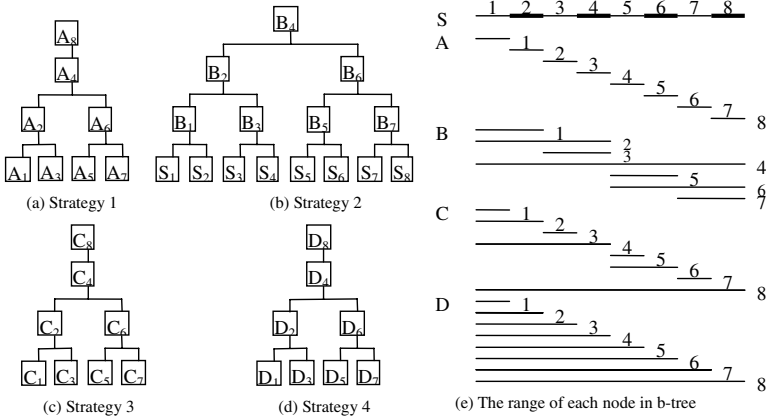
1. Initialize sketch  $RS$  to 0.
2. Search points/regions to construct  $qr$ .
3. For each point/region in Step(1), find sketches to construct  $qt$  according to special strategy. Update  $RS$ .
4. Evoke **hCount** algorithm to get results from  $RS$ .

For example, assume that query rectangle  $qr$  is shown as shadowed part in Fig. 1(b) and query time  $qt$  is set to  $[2, 5]$ . In Step (2), we search from the root of R-tree, and find that  $R_1 \subset qr$  and  $R_2$  intersects  $qr$ ; continue to search  $R_2$ , we find that  $p_4 \in R_2$  and  $p_3$  is out of  $R_2$ . Then, in Step (3), we search B-trees affiliated to  $R_1$  and  $p_4$  to get suitable sketches to update  $RS$ .

### 3.2 Four Strategies to Reserve Sketches

Previously, we have introduced that sketches can be indexed by an sRB-tree. Here, we propose four strategies to organize sketches for each spatial region/point in Fig. 4.

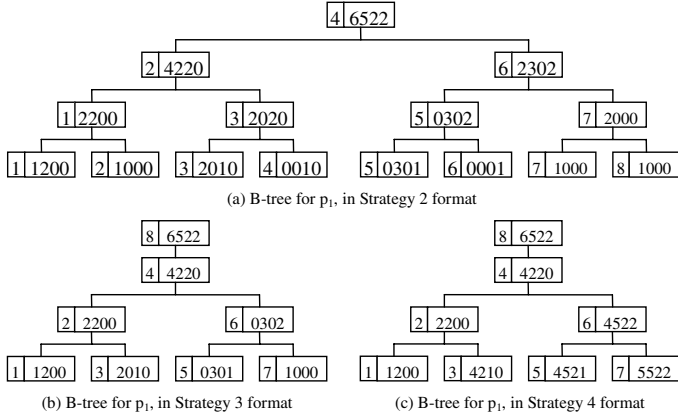
Let  $S_i$  denote the sketch produced by *sketch producer* at timestamp  $i$ .  $A_i$ ,  $B_i$ ,  $C_i$  and  $D_i$  are denoted as sketches used in four strategies, whose intervals are shown in Fig. 4(e). Fig. 5 shows B-tree constructed for  $p_1$  based on Fig. 1(c).



**Fig. 4.** Four Strategies

**Strategy 1:** The most direct solution is to reserve each sketch transmitted from Sketch Producer in database respectively, which can be found in Figure 3 and Figure 4(a).

Meanwhile, this method is not efficient to execute a query. Assume  $qt$  is  $[a, b]$ ,  $b - a + 1$  sketches ( $A_a, \dots, A_b$ ) must be chosen from the B-tree to construct result sketch. In an utter situation where  $qt = [1, T]$ , all  $T$  sketches must be found out.



**Fig. 5.** An example on Strategies

**Strategy 2:** Strategy 1 is not efficient in answering a query, especially when  $qt$  is large. Fig. 4(b) shows another way to organize sketches. The sketch in each leaf entry (denoted as  $S_i$ ) is just the sketch transmitted from sketch producers, while the sketch of each non-leaf node (denoted as  $B_t$ ) is the sum of sketches within  $[t - 2^{\text{level}(B_t)} + 1, t + 2^{\text{level}(B_t)}]$ , where  $\text{level}(B_t) = \max(i | t \bmod 2^i = 0) + 1$ . For any  $S_i$ ,  $\text{level}(S_i) = 0$ . We can find from Figure 4(b) that the number of sketches reserved in *database* is  $2T - 1$ .

*Property 1.* For any sketch  $B_t$  presenting  $[a_1, b_1]$  where  $b_1 \neq T$ , for any  $l$ ,  $0 \leq l < \text{level}(B_t)$ , we can find a sketch  $s$  presenting  $[a_2, b_2]$ ,  $\text{level}(s) = l$ ,  $b_1 + 1 = a_2$ .

For example,  $B_2$  presents  $[1, 4]$ ,  $\text{level}(B_2) = 2$ . We can find two sketches  $B_5$ ,  $S_5$ , each presenting  $[5, 6]$ ,  $[5, 5]$  and  $\text{level}(B_5) = 1$ ,  $\text{level}(S_5) = 0$ .

*Property 2.* For any sketch  $B_t$  presenting  $[a_1, b_1]$  where  $a_1 \neq 1$ , for any  $l$ ,  $0 \leq l < \text{level}(B_t)$ , we can find a sketch  $s$  presenting  $[a_2, b_2]$ ,  $\text{level}(s) = l$ ,  $a_1 - 1 = b_2$ .

For example,  $B_7$  presents  $[7, 8]$ ,  $\text{level}(B_7) = 1$ . We can find that sketch  $S_6$  presents  $[6, 6]$ ,  $\text{level}(S_6) = 0$ .

These two properties can be applied to construct result sketch for  $qt$  like  $[a, b]$ . Aware that  $[a, b] = [1, b] - [1, a - 1]$  where  $a > 1$ , the problem can be solved if  $qt$  in format  $[1, t] (1 \leq t \leq T)$  can be constructed. First, present  $t$  in binary format; second, use two properties to select sketches. For example,  $T = 8, t = 5$ . 5 can be presented as 101 in binary format. By Property 1, we first choose  $B_2 (\text{level}(B_2) = 2, \text{its interval begins in } 1)$ , and then, choose  $S_5 (\text{level}(S_5) = 0, \text{its interval begins in } 5)$ . The result sketch can be constructed as  $B_2 + S_5$ . The number of sketches is equal to the number of '1' bits. By Property 2, we first choose  $B_4 (\text{the largest interval, presenting } [1, T])$ , and then,  $B_7 (\text{level}(B_7) = 1, \text{presenting } [7, T])$ , and finally,  $S_6 (\text{level}(S_6) = 0, \text{presenting } [6, 6])$ . The result sketch can be constructed by  $B_4 - B_7 - S_6$ . The number of sketches is equal

to the number of '0' bits plus 2. Alternative property is chosen to decrease the number of sketches. Clearly, arbitrary interval  $[a, b]$  can be constructed by at most  $\log T + 1$  sketches.

**Strategy 3:** Fig. 4(c) shows another strategy. Each sketch  $C_t$  in this strategy (Strategy 3) presents the sum of sketches within  $[t - 2^{\text{level}(t)} + 1, t]$ , where  $\text{level}(t) = \max(i | t \bmod 2^i = 0)$ . Strategy 3 also owns Property 1 of Strategy 2, so that an interval like  $[1, t]$  can be constructed by at most  $\log T$  sketches. Similarly, any interval  $[a, b]$  can be constructed by no more than  $2 \log T$  sketches.

**Strategy 4:** The last strategy (Strategy 4) is shown in Fig. 4(d). Each sketch  $D_t$  in this strategy presents the sum of sketches within  $[1, t]$ .

When sketches are organized by this strategy, queries can be executed quickly. An arbitrary interval  $[a, b]$  can be constructed by  $D_b - D_{a-1}$  when  $a > 1$ , or just  $D_b$  when  $a = 1$ . At most 2 sketches is enough.

### 3.3 Discussion

**Performance Analysis.** Table 1 summarizes the performance of four strategies. We can observe that the space requirement in Strategy 2 is nearly 2 times of other three. When sketches are organized in Strategy 4, an arbitrary interval can be constructed in shortest time.

**Table. 1.** Performance summary of four strategies

	Strategy 1	Strategy 2	Strategy 3	Strategy 4
sketches reserved	$T$	$2T - 1$	$T$	$T$
sketches accessed at most	$T$	$\log T + 1$	$2 \log T$	2

**Extension.** Currently, various kinds of sketches are developed for aggregate query from a mass dataset, such as FM sketch for counting distinct items, hCount sketch and Count sketch[4] for mining frequent items, RSS(random subset-sums)[10] sketch for quantile, and tug-of-war sketch[2] for  $F_2$  moment, etc. These sketches can be divided into two groups. The first group can only support *plus* set operator, which means that we can create the sketch for  $D_1 + D_2$  based on  $s_1$  and  $s_2$  where  $s_1, s_2$  are sketches created for dataset  $D_1, D_2$  and  $D_1 \cap D_2 = \emptyset$ . Besides of it, the second group can also support *minus* set operator, which means that we can calculate a sketch for  $D_1 - D_2$  when  $D_1 \supset D_2$  based on  $s_1$  and  $s_2$ . FM sketch belongs to the first group. Along with hCount sketch, Count sketch, RSS sketch and tug-of-war sketch belong to the second group. All of these sketches can also be integrated with sRB-tree to solve special problem in spatio-temporal applications. The first group can only organize sketches in Strategy 1 and 3. Meanwhile, all four strategies can be applied to organize sketches for the second group.



## 4 Experiments

In this section, we will conduct some experiments to study the performance of our method. Sect 4.1 examines the cost in executing a query, while in Sect 4.2, we focus on the quality of query result.

### 4.1 Query Performance

Our query method shows that the time to execute a query mainly consists of two parts: constructing result sketch  $RS$  and invoking hCount algorithm on  $RS$ . The cost in invoking hCount algorithm is acceptable when the range of the universe  $M$  is not large, which has been well studied in [12]. Here, we focus on the cost to build a result sketch. Because sketches are reserved in disk, the cost to build result sketch is linear to the number of sketches fetched, which is equal to the number of R-tree entries selected(Step 2 in Query Framework) times the number of sketches fetched per B-tree. Due to lack of real spatio-temporal datasets, we implement our experiments in two steps. First, we construct R-tree based on a real spatial dataset(downloaded from <http://www.rtreeportal.org/spatial.html>) containing 9,203 culture landmarks in North American. Figure 6(a) shows that the number of R-tree entries selected increase when  $qr$  varies from 5% to 25% of the maximal region. Second, we can build B-tree(the maximal time is 512) according to four strategies, and test the number of sketches required to construct any  $qt$ . Figure 6(b) illustrates the number of sketches required in different strategies when the query time interval  $qt$  increases from 2 to 256. Strategy 4 behaves best, only 2 sketches are wanted whenever. Strategy 1 is the worst one, in which the number of sketches is equal to the length of  $qt$ . The number of sketches also increases in Strategy 2,3, but the speed is quite slow.

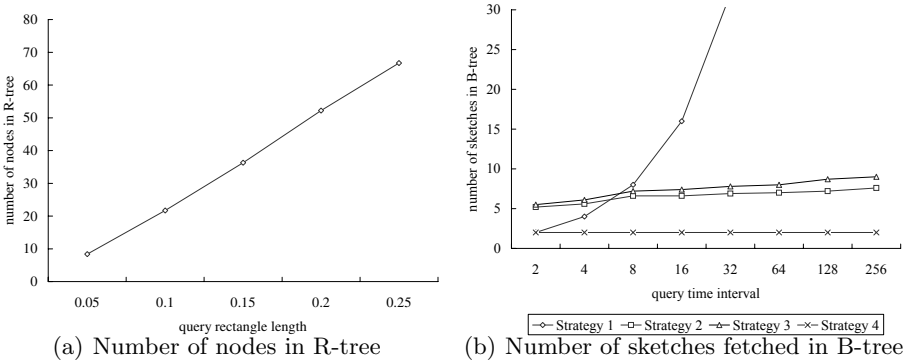


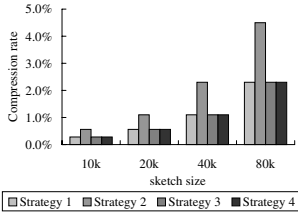
Fig. 6. Query Performance

## 4.2 Compression Rate

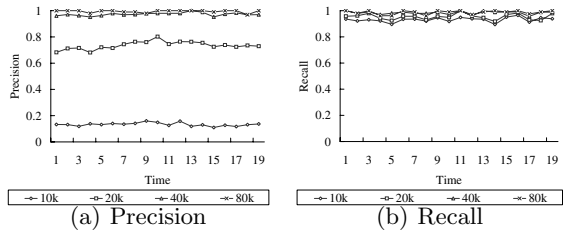
We have claimed earlier that our algorithm can present a mass dataset by a small space, while error of query result can also be guaranteed. In this section, we will show it through experiments. The topic detection and tracking (TDT) is an important issue in information retrieval and text mining (<http://www.1dc.upenn.edu/Projects/TDT3/>). We obtained a news collection of the news stories distributed through Reuters real-time datafeed, which covers 114 days and 100,672,866 (duplicate) words. We removed all articles such as “the” and “a” and preprocessed the data collection by term stemming. This dataset can be simulated as items for one point in a spatio-temporal application. Words of articles in  $t$  day can be treated as items created at timestamp  $t$ . We organize such a dataset in four strategies.

Figure 7 compares compression rate among strategies. The x-axis is the size of the a single sketch, ranged from 10k to 80k, and y-axis is the compression rate. In all situations, the compression rates are no more than 5%. We can also observe that the compression rate of Strategy 2 is nearly two times of other 3 strategies.

*recall* and *precision* are two important factors to evaluate query result. The *recall* of a result is the proportion of the frequent items that are found by the method. The *precision* is the proportion of items identified by the algorithm which are frequent items. We conduct extensive testing to show the performance with different sketch size. We separate the whole time into a serial of continuous query time intervals, whose length is 6. For each  $qt$ , output items over threshold 0.2%. Figure 6 shows precision and recall with different sketch size. Clearly, when sketch size is over 40k, the algorithm can obtain high precision and recall at the same time.



**Fig. 7.** Compression rate of four strategies



**Fig. 8.** Precision and recall under different sketches size

## 5 Conclusion

Efficient aggregation is the critical goal for most spatio-temporal applications. Recent work mostly focuses on how to count distinct moving objects and how to mine association rules in spatio-temporal applications. Rare work done on mining frequent items, which is also an important topic in various fields, including

spatio-temporal applications. Although this problem can be solved by executing queries upon a database where all items are reserved one by one, it is not efficient neither in space requirement, nor in querying response time.

In this paper, we first define the problem (FI-query) of mining frequent items in spatio-temporal applications formally. And then, we propose a novel approach for this problem, where **hCount** sketch is used to compress a mass dataset and sRB-tree is applied to index sketches. Aware that **hCount** sketch can support *plus* and *minus* set operators, B-tree of sRB-tree can be constructed by four strategies. Experiments show that this method behaves well for FI-query problem. Other sketches with similar characters to **hCount** sketch can also be integrated with sRB-tree for spatio-temporal applications. For example, we can integrate RSS sketch and sRB-tree to solve quantile problem in spatio-temporal applications.

## References

1. R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In *Proc. of VLDB*, 1994.
2. N. Alon, Y. Matias, and M. Szegedy. The space complexity of approximating the frequency moments. In *Proc. of ACM STOC*, 1996.
3. N. Beckmann, H. Kriegel, R. Schneider, and B. Seeger. The r\*-tree: An efficient and robust access method for points and rectangles. In *Proc. of ACM SIGMOD*, 1990.
4. M. Charikar, K. Chen, and M. Farach-Colton. Finding frequent items in data streams. In *Proc. of the 29th ICALP*, 2002.
5. G. Cormode and S. Muthukrishnan. What's hot and what's not: Tracking most frequent items dynamically. In *Proc. of ACM STOC*, 2003.
6. E. Demaine, A. López-Ortiz, and J. I. Munro. Frequency estimation of internet packet streams with limited space. In *Proc. of 10th Annual European Symposium on Algorithms*, 2002.
7. C. Estan and G. Verghese. New directions in traffic measurement and accounting. In *ACM SIGCOMM Internet measurement workshop*, 2001.
8. M. Fang, N. Shivakumar, H. Garcla-Molina, R. Motwani, and J. Ullman. Computing iceberg queries efficiently. In *Proc. of VLDB*, 1998.
9. P. Flajolet and G. Martin. Probabilistic counting algorithms for data base applications. *JCSS*, 2(32):182–209, 1985.
10. A. Gilbert, Y. Kotidis, S. Muthukrishnan, and M. Strauss. How to summarize the universe: Dynamic maintenance of quantiles. In *Proc. of VLDB*, 2002.
11. A. Guttman. R-trees: A dynamic index structure for spatial searching. In *Proc. of ACM SIGMOD*, 1984.
12. C. Jin, W. Qian, C. Sha, J. X. Yu, and A. Zhou. Dynamically maintaining frequent items over a data stream. In *Proc. of ACM CIKM*, 2003.
13. N. Kline and R. Snodgrass. Computing temporal aggregates. In *Proc. of ICDE*, 1995.
14. G. S. Manku and R. Motwani. Approximate frequency counts over data streams. In *Proc. of VLDB*, 2002.
15. D. Papadias, Y. Tao, P. Kalnis, and J. Zhang. Indexing spatio-temporal data warehouses. In *Proc. of ICDE*, 2002.
16. Y. Tao, G. Kollios, J. Considine, F. Li, and D. Papadias. Spatio-temporal aggregation using sketches. In *Proc. of ICDE*, 2004.

# Discovery of Frequent XML Query Patterns with DTD Cardinality Constraints\*

Yunfeng Liu, Dongqing Yang, Shiwei Tang, Tengjiao Wang, and Jun Gao

School of Electronics Engineering and Computer Science  
Peking University, Beijing, China  
yfliu@db.pku.edu.cn

**Abstract.** Common query patterns of multiple XML queries can be stored and shared to accelerate the query execution efficiently. We present a new technique for efficiently mining frequent XML query patterns with DTD cardinality constraints. We first introduce a new definition of extended subtree for XML query patterns (EST), which provides a model for supporting special characters of “common” XML query pattern. Then we “push” the DTD cardinality constraints deep into the mining process to prune the search space and still ensure the completeness of the answers. Finally, we propose an algorithm FEST for effectively mining frequent ESTs to improve the query performance for XML data management system.

## 1 Introduction

The problem of efficiently answering XML queries has recently received increasing attention by the research community. Many efforts have been devoted to define query optimization techniques[1,2,3,4,5]. In the multi-query optimization, the common query patterns of multiple XML queries can be stored and shared to accelerate the query execution efficiently. Such common patterns typically arise in many applications like XML materialized view selection and semantic caching system for XML queries [6,7,8,9].

But how to exploit the frequent common patterns is a difficult problem. Previous works devoted to merge the multiple XML queries to generate the common expressions, but this needs a lot of expensive calculations of query containment and equivalence for XPath fragments, which is a CONP-Complete problem[2,3,4].

In this paper we present a new technique for efficiently mining frequent XML query patterns with DTD cardinality constraints and examine how the query performance in XML data management system can be improved effectively by using these frequent tree patterns. More specifically, we make the following contributions:

---

\* This work is supported by the NKBRSF of China (973) under grant No.G1999032705, the National ‘863’ High-Tech Program of China under grant No. 2002AA4Z3440.

- We introduce a new definition of extended subtree for XML query patterns (EST), which provides a model for supporting special characters of “common” XML query pattern such as the wildcard “\*”, relative path “/” and the recursion. EST can extract patterns “hidden” (or embedded) deep within multiple query trees which might be missed by the traditional definition of subtree.
- We “push” the DTD cardinality constraints deep into the mining process to prune the search space and still ensure the completeness of the answers.
- Using the anti-monotone property, we propose an algorithm FEST for effectively mining frequent ESTs to improve the query performance for XML data management system.

The theoretical analysis shows that pruning the search space using DTD constraints can still ensure the completeness of the answers. We conduct a set of experiments in our CoXML system and the experimental results generated from the real data reveal that the algorithm works well in practice.

## 2 Preliminary Concepts and Problem Statement

We introduce a new definition of extended subtree[10] to describe XML query pattern (EST) here, which provides a model for supporting special characters of “common” XML query pattern such as the wildcard “\*”, relative path “/” and the recursion and can extract patterns “hidden” (or embedded) deep within large trees which might be missed by the traditional definition.

We adopt a string representation of a tree in [10] and use the notation  $l(T)$  to refer to the label sequence of  $T$ , which consists of the node labels of  $T$  in depth-first ordering.

**Definition 1. (EST: Extended Subtree for XML Query Pattern)** We say that a tree  $S = (N_s, B_s)$  is an EST of  $T = (N, B)$ , denoted as  $S \equiv T$ , provided i)  $N_s \subseteq N$ , ii)  $b = (n_x, n_y) \in B_s$  if and only if  $n_x \preceq_l n_y$ , i.e.,  $n_x$  is an ancestor of  $n_y$  in  $T$ . In other words, we require that a branch appears in  $S$  if and only if the two vertices are on the same path from the root to a leaf in  $T$ . If  $S \equiv T$ , we also say that  $T$  contains  $S$ . An EST of size  $k$  is also called a  $k$ -subtree or  $k$ -EST.

**Definition 2. (Scope for Tree)** Let  $T(n_l)$  refer to the tree rooted at node  $n_l$ , and let  $n_r$  be the right-most leaf node in  $T(n_l)$ . The scope of node  $n_l$  is given as the interval  $[l, r]$ , i.e., the lower bound is the position ( $l$ ) of node  $n_l$ , and the upper bound is the position ( $r$ ) of node  $n_r$ . The concept of scope will play an important part in counting subtree frequency.

Let  $D$  denote a database of trees (i.e., a forest), and let subtree  $S \equiv T$  for some  $T \in D$ . Each occurrence of  $S$  can be identified by its match label, which is given as the set of matching positions (in  $T$ ) for nodes in  $S$ .

Let  $\delta_T(S)$  denote the number of occurrences of the subtree  $S$  in a tree  $T$ . Let  $d_T(S) = 1$  if  $\delta_T(S) > 0$  and  $d_T(S) = 0$  if  $\delta_T(S) = 0$ . To find the “frequent” query patterns, we use the weighted support as the condition of mining. The weighted support of  $S$  is defined as  $\sigma_w(S) = \sum_{T \in D} \delta_T(S)$ , i.e., total number of occurrences of  $S$  over all trees in  $D$ . Typically, support is given as a percentage of the total number of trees in  $D$ .

**Definition 3. (Frequent EST Mining Problem)** Given a user specified weighted support value  $W\_Sup$ , the frequent EST mining problem is to efficiently enumerate all frequent EST in  $D$  whose occurrence frequency is greater than or equal to the  $W\_Sup$  threshold.

XML query trees, specified over a set of XML documents, usually conform to a Document Type Definition (DTD) (or an equivalent, such as XML Schema). We will use cardinality constraints in DTD[11] to make the mining process more efficient and still ensure the completeness of the answers.

**Definition 4. (Cardinality Constraints)** In a DTD declaration, there are 4 possible cardinality relationships between an element and its sub-elements as illustrated below:

<!ELEMENT article (title, author+, ref\*, price?)>

- 1) (0,1): An element can have either zero or one sub-element.
- 2) (1,1): An element must have one and only one sub-element.
- 3) (0,N): An element can have zero or more sub-elements.
- 4) (1,N): An element can have one or more sub-elements.

For convenience, we call each cardinality relationship as type A, B, C, D, respectively.

**Definition 5. (DTD Cardinality Constraints Graph (CCG))**  $G$  is a pair  $(V, E)$ , where  $V$  is a finite set and  $E$  is a binary relation on  $V$ . The set  $V$  consists of element and attributes in a DTD. Each edge  $e \in E$ , is labeled with the cardinality constraints relationships.

### 3 Mining Frequent XML Query Patterns with DTD Cardinality Constraints

We present a technique which can effectively mine frequent XML query patterns with DTD cardinality constraints. We “push” the DTD constraints deep into the whole mining process to prune the search space and still ensure the completeness of the answers.

#### 3.1 Candidate Representation

The concept of an equivalence class has proved very useful for systematic candidate generation for itemsets and sequences [12]. We now extend this concept to ESTs as well.

**Definition 6. (Equivalence Class for ESTs)** We say that two  $k$ -ESTs  $X, Y$  are in the same prefix equivalence class iff they share a common prefix up to the  $(k - 1)$ th node. Formally, let  $x, y$  be the string encodings of two trees, and let function  $p(x, i)$  return the prefix up to the  $i$ th node.  $X, Y$  are in the same class iff  $p(x, k - 1) = p(y, k - 1)$ . Thus any two members of an equivalence class differ only in the position of the last node.

We use a rightmost branch expansion technique to grow a tree by attaching new nodes only on the rightmost branch of the tree. It is sufficient to implement incremental computation. To find frequent  $k$ -itemsets, a set of candidate  $k$ -itemsets is generated by self joining of the scope-list of  $(k-1)$ -itemsets.

We use the notation  $L(X)$  to refer to the scope-list [10] of a tree  $X$ . Each element of the scope-list is a triple  $(t, m, s)$ , where  $t$  is a tree id (tid) in which  $X$  occurs,  $m$  is a match label of the  $(k - 1)$  length prefix of  $X$ , and  $s$  is the scope of the last item  $x_k$ . So the  $L(X)$  shows the information that where EST  $X$  occurs and the frequency of  $X$  occurs. If a scope-list of  $X$  occurs in at least the value of the minimum weighted support, the pattern is considered frequent.

Let  $l_1(EST1)$  and  $l_2(EST2)$  be the scope-list of two  $(k-1)$ -ESTs in  $L_{k-1}$ ,  $l_i[j]$  be the  $j$ th item in  $l_i$  and  $l_i[j]$  be the triple of  $(t_i, m_i, s_i)$ . The self join of  $L_{k-1}$  and  $L_{k-1}$  is performed where member of  $L_{k-1}$  are joinable if they satisfy the following 2 conditions: (1)  $t_i = t_j$ , (2) the first  $(k-2)$  items of  $m_i$  and  $m_j$  are in common. If  $s_i \subset s_j$ , then according to the notion of the rightmost branch expansion, add  $s_i$  as the child of  $s_j$ . If  $s_i \subset s_j$ , then add  $s_j$  as the sibling of  $s_i$ .

3.2 Using Cardinality Constraints to Reduce Candidate Generation Space

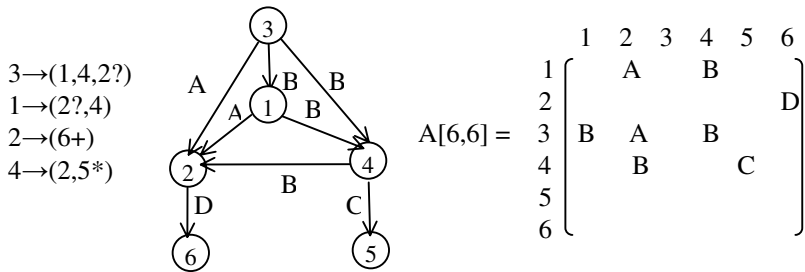
DTD-graph  $G=(V, E)$  is a directed graph and we use the labeled adjacent matrix representation. Suppose the set of labels of  $G$  is  $L=\{1,2,...,n\}$ , the adjacent matrix for  $G$  is an  $n \times n$  matrix  $A$ , where  $A[i, j]$  is one of the cardinality constraints relationships, that is the label of the arc from vertex  $i$  to vertex  $j$  in  $G$ . And a blank represents the absence of an arc.

**Example 1.** Now consider a simple DTD and its CCG in Fig. 1. The adjacent matrix shown in the right of Fig. 1 is the cardinality constraints between adjacent pairs.

In order to get cardinality constraints relationships between all pairs in CCG, we define a function :  $F(x, y)=x \cup_+ y$ ,  $x, y \in \{A, B, C, D\}$ . It is defined as follows:

	$A \cup_+ A$	$A \cup_+ B$	$A \cup_+ C$	$A \cup_+ D$	$B \cup_+ B$	$B \cup_+ C$	$B \cup_+ D$	$C \cup_+ C$	$C \cup_+ D$	$D \cup_+ D$
$F(x, y)$	A	A	C	C	B	C	D	C	C	D

Table 1 describes algorithm GetCardinalityConstraint for computing the cardinality constraints between all pairs in CCG. Now we can use the adjacent matrix for cardinality constraints between all pairs to prune the search space during the process of class extension, including self extension according to the anti-monotone property. In other words, if an EST does not satisfy this constraint, then none of its supersets can satisfy the constraint either.



**Fig. 1.** A simple DTD D, its CCG and the adjacent matrix for cardinality constraints between adjacent pairs

**Table 1.** Algorithm for computing cardinality constraints between all pairs

GetCardinalityConstraint (A: array of [1..n, 1..n] ;A': array of [1..n, 1..n]);	
1)	int i, j, k;
2)	For i=1 to n do
3)	For j=1 to n do
4)	A'[i, j]=A[i, j];
5)	For i=1 to n do
6)	For j=1 to n do
7)	For k=1 to n do
8)	F(x, y)= A'[i, k] $\cup_+$ A'[k, j]
9)	A'[i, j]= F(x, y);

Using the algorithm GetCardinalityConstraint and the data in example 1, we can get the adjacent matrix for cardinality constraints between all pairs shown in Fig. 2.

$$A'[6,6] = \begin{matrix} & \begin{matrix} 1 & 2 & 3 & 4 & 5 & 6 \end{matrix} \\ \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{matrix} & \begin{bmatrix} & A & & B & C & C \\ & & & & & D \\ B & A & & B & C & C \\ & B & & & C & D \\ & & & & & \\ & & & & & \end{bmatrix} \end{matrix}$$

**Fig. 2.** Adjacent matrix for cardinality constraints between all pairs

Pruning by anti-monotone property can be applied at each iteration of Apriori-style algorithms to help improve the efficiency of the overall mining process, while guaranteeing completeness of the data mining query response.



### 3.3 FEST Algorithm for Mining Frequent ESTs

In this section, we present the whole algorithm for effectively mining frequent ESTs with DTD cardinality constraints. There are two main steps to enumerate all the frequent ESTs that occur in  $D$ . First, we need a systematic way of generating candidate ESTs whose frequency is to be computed. The candidate set should be non redundant, i.e., each EST should be generated at most once. Second, we need efficient ways of counting the number of occurrences of each candidate in the database  $D$ , and to determine which candidates pass the weighted support threshold.

During the mining process, we use the adjacent matrix for cardinality constraints between all pairs to prune the search space and still ensure the completeness of the answers. Table 2 describes the whole FEST algorithm.

**Table 2.** Algorithm FEST for mining frequent ESTs

Input:  $D$  (Database of XML Query Pattern Trees),  $M$  (Matrix for DTD Cardinality Constraints) and  $W\_Sup$  (Minimum Weighted Support)

Output:  $L$ , Frequent Subtrees in  $D$

```

1)  $L_1 = \text{Get\_Frequent\_Node}(D, W\_Sup)$  /*get the frequent 1-node subtrees in  $D$ */;
2) for ( $k = 2$ ;  $L_{k-1} \neq \emptyset$ ;  $k++$ )
3) {
4)    $C_k = \text{Get\_Candidate\_ESTs}(L_{k-1}, M)$ ;
5)   for each subtree  $s \in C_k$ 
6)     if ( $\text{GetScopeListLenth}(s) > W\_Sup$ ) then
7)        $L_k = L_k \cup \{s\}$ ;
8) }
9) return  $L = \cup_k L_k$ ;

procedure  $\text{Get\_Candidate\_ESTs}(L_{k-1}$ : frequent  $(k-1)$ -itemset;  $M$ : Matrix for DTD Cardinality Constraints)
1) for each subtree  $l_1 \in L_{k-1}$ 
2)   for each subtree  $l_2 \in L_{k-1}$ 
3)     if ( $l_1.\text{Prefix} = l_2.\text{Prefix}$ ) then
4)       {
5)          $R = \text{RightmostBranchExpansion}(l_1, l_2)$ ;
6)         for each subtree  $l_e \in R$ 
7)           if ( $\text{ValidateCardinalityConstraints}(l_e, M) = \text{TRUE}$ ) then
8)             {
9)                $c = \text{GetSelfJoin}(l_1, l_2, l_e)$ ;
10)               $C_k = C_k \cup \{c\}$ ;
11)            }
12)          }
13)   return  $C_k$ ;

```

**Example 2.** Fig. 3 shows a database of 3 query trees conforming to the DTD in Fig. 1, The user specified minimal weighted support  $W\_Sup$  is 3. Fig. 4 shows the mining result  $F_4$  of all the frequent ESTs in the database using algorithm FEST. Table 3 and Table 4 show the main process of mining the frequent ESTs using algorithm FEST.

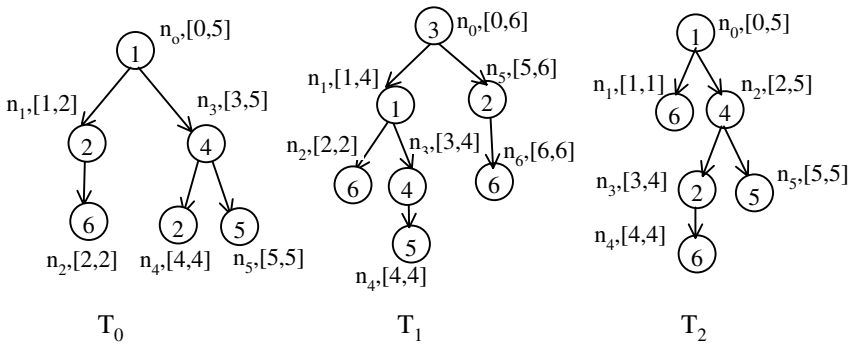


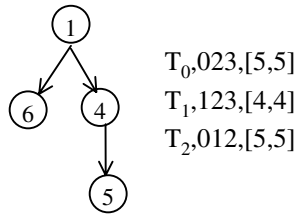
Fig. 3. Query trees conforming to the DTD in Fig. 1

Table 3. Computing  $F_1$

①	②	③	④	⑤	⑥
$T_0, 0, [0,5]$ $T_1, 1, [1,4]$ $T_2, 0, [0,5]$	$T_0, 1, [1,2]$ $T_0, 4, [4,4]$ $T_1, 5, [5,6]$ $T_2, 3, [3,4]$	$T_1, 0, [0,6]$	$T_0, 3, [3,5]$ $T_1, 3, [3,4]$ $T_2, 2, [2,5]$	$T_0, 5, [5,5]$ $T_1, 4, [4,4]$ $T_2, 5, [5,5]$	$T_0, 2, [2,2]$ $T_1, 2, [2,2]$ $T_1, 6, [6,6]$ $T_2, 1, [1,1]$ $T_2, 4, [4,4]$

Table 4. Computing  $F_2$  (partial)

$\begin{matrix} \textcircled{1} \\ \downarrow \\ \textcircled{1} \end{matrix}$	$\begin{matrix} \textcircled{1} \\ \downarrow \\ \textcircled{2} \end{matrix}$	$\begin{matrix} \textcircled{1} \\ \downarrow \\ \textcircled{4} \end{matrix}$	$\begin{matrix} \textcircled{1} \\ \downarrow \\ \textcircled{5} \end{matrix}$	$\begin{matrix} \textcircled{1} \\ \downarrow \\ \textcircled{6} \end{matrix}$	$\begin{matrix} \textcircled{2} \\ \downarrow \\ \textcircled{1} \end{matrix}$	$\begin{matrix} \textcircled{2} \\ \downarrow \\ \textcircled{2} \end{matrix}$
×	$T_0, 0, [1,2]$ $T_0, 0, [4,4]$ $T_2, 0, [3,4]$	$T_0, 0, [3,5]$ $T_1, 1, [3,4]$ $T_2, 0, [2,5]$	$T_0, 0, [5,5]$ $T_1, 1, [4,4]$ $T_2, 0, [5,5]$	$T_0, 0, [2,2]$ $T_1, 1, [2,2]$ $T_2, 0, [1,1]$ $T_2, 0, [4,4]$	×	×
$\begin{matrix} \textcircled{2} \\ \downarrow \\ \textcircled{5} \end{matrix}$	$\begin{matrix} \textcircled{2} \\ \downarrow \\ \textcircled{6} \end{matrix}$	$\begin{matrix} \textcircled{4} \\ \downarrow \\ \textcircled{1} \end{matrix}$	$\begin{matrix} \textcircled{4} \\ \downarrow \\ \textcircled{2} \end{matrix}$	$\begin{matrix} \textcircled{4} \\ \downarrow \\ \textcircled{4} \end{matrix}$	$\begin{matrix} \textcircled{4} \\ \downarrow \\ \textcircled{5} \end{matrix}$	$\begin{matrix} \textcircled{4} \\ \downarrow \\ \textcircled{6} \end{matrix}$
×	$T_0, 1, [2,2]$ $T_1, 5, [6,6]$ $T_2, 3, [4,4]$	×	$T_0, 3, [4,4]$ $T_2, 2, [3,4]$	×	$T_0, 0, [5,5]$ $T_1, 1, [4,4]$ $T_2, 0, [5,5]$	$T_2, 2, [4,4]$



**Fig. 4.** Result of  $F_4$

Computing subsequent frequent patterns in the same way until no new candidate is generated. Here, we only show the result  $F_4$  because of the limited space.

During the mining process, in order to count the number of occurrences of each generated extended subtree, we can scan the scope-list of them. If the number of the triples of a subtree is more than or equal to a user-specified threshold, then the subtree is frequent, and otherwise, it should be eliminated from the candidate ESTs. On the other hand, we use the adjacent matrix for cardinality constraints between all pairs to prune the search space (with symbol ‘ $\times$ ’).

## 4 Analysis of Algorithm and Experimental Evaluation

**Lemma 1.** For every EST  $P$  in equivalence class of  $k$ -ESTs, let  $n_r$  is the right-most leaf node in  $P$ , whose scope is given as  $[r,r]$ . Then valid element may be attached to only those nodes that lie on the path from the root to the right-most leaf  $n_r$  in  $P$  for generating  $(k+1)$ -ESTs.

The key of the candidate ESTs generation in the FEST algorithm is the rightmost branch expansion, a technique to grow a tree by attaching new nodes only on the rightmost branch of the tree.

**Theorem 1. (Soundness of the Algorithms)** Every tree in the  $L = \cup_k L_k$  generated by FEST algorithm is a frequent EST in  $D$  with the occurrence frequency greater than or equal to the weighted support threshold.

**Theorem 2. (Completeness of the Algorithm)** The algorithm FEST generates all possible frequent ESTs.

Recently, we are developing CoXML, an XML-based information integration system. We have conducted a series of experiments in CoXML system to test the efficiency of the algorithm and compare it with the tree mining algorithm without DTD constraints. We can see that the PEST algorithm achieves higher performance.

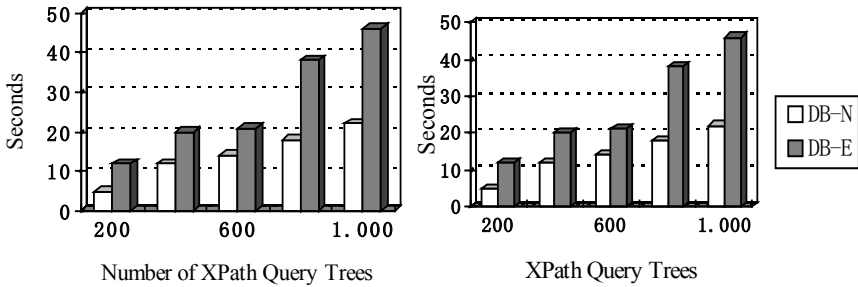
### • The First Experiment: Efficiency of FEST Algorithm Running on Different Data Sets

In this experiment, we check the efficiency of FEST algorithm running on different data sets: one for e-business DB-E and the other for news DB-N. Fig. 5(a) depicts the execution time of the FEST algorithm with weighted support varying from 1% to 5%. Fig. 5(b) depicts the execution time with a number of XPath query trees varying from 200 to 1000.

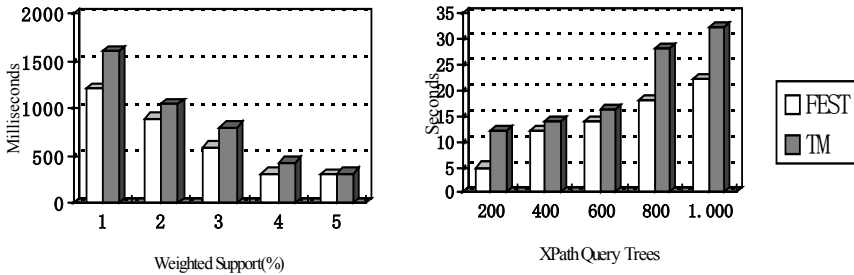
- The Second Experiment: **Comparing the FEST Algorithm against the Tree Mining Algorithm without DTD Constraints**

Now, we compare the FEST algorithm against the Tree Mining algorithm (TM) without DTD constraints according to the weighted support and the numbers of XPath query trees. Fig. 6(a) depicts the execution time of the two algorithm with weighted support varying from 1% to 5%. Fig. 6(b) depicts the execution time of the algorithm with a number of XPath query trees varying from 200 to 1000. These two were conducted both on DB-N.

The set of results conducted on the real XML data reveal that the algorithms proposed in the paper work well in practice.



**Fig. 5.** (a) Varying weighted support. (b) Varying the number of XPath query trees



**Fig. 6.** (a) Varying weighted support. (b) Varying the number of query trees

## 5 Conclusion

We present a new technique for efficiently mining frequent XML query patterns with DTD cardinality constraints. First we introduce a new definition of extended subtree for XML query patterns (EST). Then we “push” the DTD cardinality constraints deep into the mining process to prune the search space and still ensure the completeness of the answers. At last, using the anti-monotone property, we propose an algorithm FEST for effectively mining frequent ESTs to improve the query performance for

XML data management system. We conduct a set of experiments in our CoXML system. The experimental results generated from the real data reveal that the algorithms work well in practice.

## References

1. G. Gottlob, C. Koch, R. Pichler, Efficient Algorithms for Processing XPath queries, In Proceedings of the 28th International Conference on Very Large Data Bases (VLDB'02) Hong Kong, China August 20-23, 2002.
2. Quanzhong Li and Bongki Moon. Indexing and Querying XML Data for Regular Path Expressions. In Proceedings of the 27th International Conference on Very Large Data Bases (VLDB'01), pages 361--370, Rome, Italy, September 2001.
3. G. Miklau, D. Suciu, Containment and Equivalence for an XPath Fragment, In Proceedings of the Twenty-first ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS'02), 2002.
4. P. T. Wood, Containment for XPath Fragments under DTD Constraints, In Proceedings of the 9th International Conference on Database Theory, Siena, Italy, January 8-10, 2003.
5. S. Amer-Yahia, S. Cho, L. K. S. Lakshmanan, D. Srivastava, Minimization of tree pattern queries, In Proceedings of the ACM SIGMOD International Conference on Management of Data, Santa Barbara, California, USA, May 21-24, 2001.
6. Xin Zhang, Katica Dimitrova, etc. RainbowII: Multi-XQuery Optimization Using Materialized XML Views. In Proceedings of the ACM SIGMOD International Conference on Management of Data. San Diego, California, USA. 2003..
7. Katica Dimitrova, M. EL-Sayed and E. Rundensteiner. Order-sensitive View Maintenance of Materialized XQuery Views, In Proceedings of the International Conference on Conceptual Modeling(ER'03). Chicago, Illinois, 2003.
8. B. Chidlovskii, U. M. Borghoff. Semantic Caching of Web Queries. VLDB Journal 9(1): 2-17, 2000.
9. Liang Huai Yang, Mong Li Lee, Wynne Hsu. Efficient Mining of XML Query Patterns for Caching. In Proceedings of the 29th International Conference on Very Large Data Bases (VLDB'03) Berlin, Germany, 2003.
10. Mohammed J. Zaki. Efficiently Mining Frequent Trees in a Forest. In Proceedings of the ACM SIGKDD International Conference, Edmonton, Alberta, Canada, 2002.
11. D. Lee and W. W. Chu. Constraints-preserving Transformation from XML Document Type Definition to Relational Schema. In Proceedings of the 19th International Conference on Conceptual Modeling(ER'00). Springer-Verlag, Salt Lake City, UT, Oct. 2000.
12. M. J. Zaki. Scalable Algorithms for Association Mining. IEEE Transactions on Knowledge and Data Engineering, 12(3):372-390, May-June 2000.

# Mining Inheritance Rules from Genealogical Data

Yen-Liang Chen and Jing-Tin Lu

Dept. of Information Management, National Central Univ, Chung-Li, Taiwan 320  
ylchen@mgt.ncu.edu.tw

**Abstract.** Data mining extracts implicit, previously unknown and potentially useful information from databases. Many approaches have been proposed to extract information, and one of the most important ones is finding association rules. Although a large number of researches have been devoted to this subject, to the best of our knowledge, no previous researches find association rules from genealogical data. In this paper, we use a DAG (directed acyclic graph) to represent the genealogical data of families, where a node can be viewed as a member in the family tree, the features associated with a node as the characteristics of the corresponding person and the arcs as the parental relationships between members. In the DAG, two kinds of inheritance rules are defined, which indicate how the characteristics of ancestors are passed down to descendants, and an algorithm containing four stages is proposed to discover the inheritance rules.

## 1 Introduction

The aim of data mining is to extract implicit, previously unknown and potentially useful information from databases [1]. Many approaches have been proposed to extract information. One of the most important ones is mining association rules, which is to find the relationship between items in transaction databases.

The methodology was first introduced by Agrawal, Imielinski and Swami [2] and can be stated as follows. Given two non-overlapping subsets of product items,  $X$  and  $Y$ , an association rule in form of  $X \rightarrow Y$  indicates a purchase pattern that if a customer purchases  $X$  then it is likely that he or she also purchases  $Y$ . Two criteria, *support* and *confidence*, are commonly used to select the association rules. Support is a measure of how often the transactional records in the database contain both  $X$  and  $Y$ , and confidence is a measure of the accuracy of the rule in predicting the purchasing pattern. By far, the Apriori algorithm [3] is the most known algorithm for mining association rules.

Since association rules are useful and easy to understand, the method has also attracted increased research interest, and many extensions have been proposed in recent years, including (1) algorithm improvements [4]; (2) fuzzy rules [5]; (3) multi-level and generalized rules [6]; (4) quantitative rules [7]; (5) spatial rules [8]; (6) inter-transaction rules [9]; (7) interesting rules [10]; (8) temporal association rules [11] and

(8) inheritance rules for flexible model retrieval [12]. Brief literature reviews of association rules are given by [1, 13].

Although a lot of extensions have been done to the area, to the best of our knowledge, no existing research has ever considered the possibility of finding association rules from a DAG (directed acyclic graph) database. The traditional association rule could only find the associations between different products, from which we could know which products a customer usually buys together. However, when we are given a database consisting of many DAG structures, the traditional method is unable to find the relations between them, because the data structure is no longer linear but becomes acyclic-graph structured. Undoubtedly, this inability means that valuable information is still hidden in the database, still unknown for users. Let us use the following example to explain the potential applications of mining rules from DAG data.

We can use a DAG to represent the genealogical data of families, where a node can be viewed as a member in a family tree, the features associated with a node as the characteristics of the corresponding person and the arcs as the parental relationships between members. The features associated with a node correspond to any observable phenomenon of that person; say the biological characteristics, social status, religion status, health conditions, types of diseases, etc. A node will have at most two incoming arcs, for a person has at most two parents. On the other hand, a node can have as many outgoing arcs as the number of children he bears. In this way, DAG data can be used to represent genealogical data consisting of many family trees. In addition, the discovered association rules may indicate how the characteristics of ancestors are passed down to descendants, and how they evolve through the inheritance process.

In this paper, the objective is to find association rules from a DAG database. However, we bound the problem in a more specific environment so that the proposed rules can have more abundant and clear semantics in practice. Here, the tradeoff is that, if we view DAG as a very generic structure, then it is difficult to define the rules that are meaningful in all kinds of situations. However, if we consider the problem in a narrower domain, then the defined rules will become more meaningful in the corresponding contexts. In view of this, we restrict our problem within the domain of mining genealogical data, from which we intend to study their inheritance relationships between generations.

Therefore, in the remainder of the paper we view a DAG as the genealogical data of families. In such a graph, two kinds of inheritance rules will be discovered, where the rules indicate how the characteristics of ancestors are passed down to descendants. In Section 2, we will formally define the problem and the inheritance rules. Next, the solution framework and data preprocessing methods will be given in Section 3. Finally, the conclusion and future perspectives are discussed in Section 4.

## 2 The Problem and the Inheritance Rules

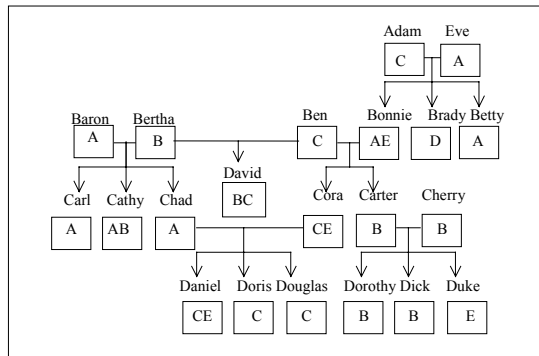
We firstly define the DAG data structure in Section 2.1. Then, Section 2.2 defines the inheritance rules.

## 2.1 The DAG Data Structure

Let  $G = (V, E)$  denote a directed acyclic graph, where  $V = \{v_1, v_2, \dots, v_n\}$  is a set of  $n$  nodes and  $E = \{e_1, e_2, \dots, e_m\}$  is a set of  $m$  arcs. Let  $Fv = \{Fv_1, Fv_2, \dots, Fv_C\}$  denote the set of all the features in the node set. And, each node  $v_x$  is associated with a set  $f_{v_x}$  of features, where  $f_{v_x} \subset Fv$ . Since a node  $v$  has at most two incoming arcs, we have  $m \leq 2 \times n$ . For ease of presentation, the following functions are defined.

1.  $V(g)$  ( $E(g)$ ) returns the set of nodes (arcs) in DAG  $g$ .
2.  $F(v_x)$  returns the features associated with node  $v_x$ .
3.  $F(V')$  returns the set of features of nodes in  $V'$ .
4.  $is-ancestor(v_x, v_y) = \text{true}$  if  $v_x$  is an ancestor of  $v_y$ ; false otherwise. If  $v_x = v_y$ , the value is true.
5.  $k-ancestor(v_x, v_y) = \text{true}$  if  $v_x$  is  $v_y$ 's ancestor  $k$  generations ago; false otherwise. The value of  $k$  must be nonnegative. When the value of  $k$  is 0, it returns true if  $v_x = v_y$ .
6.  $ancestors(v_x) = \{v_y \mid is-ancestor(v_y, v_x) = \text{true}\}$ .
7.  $descendants(v_x) = \{v_y \mid is-ancestor(v_x, v_y) = \text{true}\} - \{v_x\}$ .
8.  $T(g, ugap, lgap) = \{v_x \mid v_x \in V(g) \wedge (\exists v_y, k-ancestor(v_y, v_x) \wedge k \geq ugap) \wedge (\exists v_y, k-ancestor(v_x, v_y) \wedge k \geq lgap)\}$

In the above,  $is-ancestor(v_x, v_y)$  indicates whether  $v_x$  is an ancestor of  $v_y$ . To determine this relation, we need to check if there is a path from node  $v_x$  to node  $v_y$  in DAG. Similarly, to determine  $k-ancestor(v_x, v_y)$ , we need to check if there is a path with  $k$  arcs from  $v_x$  to  $v_y$ . As to the function  $ancestors(v_x)$ , it finds all ancestors of node  $v_x$ , including itself. On the contrary,  $descendants(v_x)$  is used to find all descendants of node  $v_x$ . Finally,  $T(g, ugap, lgap)$  denotes the set of the nodes which have ancestors at least  $ugap$  generations ago and descendants at least  $lgap$  generations later. When  $ugap=0$  and  $lgap=0$ , it equals to the set  $V(g)$ . When  $ugap=1$  and  $lgap=0$ , it will remove all the nodes who have no ancestors from  $V(g)$ . But, if we set  $ugap=2$  and  $lgap=2$ , every node in  $T(g, 2, 2)$  will have some ancestors at least two generations ago and some descendants at least two generations later. Fig. 1 shows an example of genealogical data, and Fig. 2 is the corresponding DAG.



**Fig. 1.** An example of genealogical data



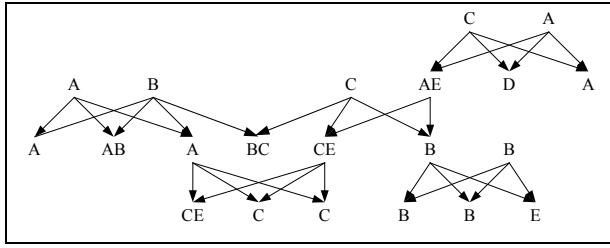


Fig. 2. The corresponding DAG

## 2.2 The Inheritance Rules

According to the definitions given in the previous section, we are ready to define the two kinds of inheritance rules. In the following, we will introduce the inheritance rules one by one, and for each one we first discuss its application scenario and then its definition.

As to the first kind, let us consider the following scenario. Suppose we already know John possesses some features, say A and B. Then, we may be interested in finding how likely the descendants of John will have some other features, say B and C. In the rule form, it can be expressed as  $AB \rightarrow BC$ . For reference, we call this kind of rule as father-descendants rule (F-D rule), because it analyzes how the features of a person pass down to his descendants.

Intuitively, the rule  $AB \rightarrow BC$  can be determined as follows. First, we must find how many fathers contain features AB in all pairs of fathers and descendants. Let this value be 4%. Next, we must find how many pairs of fathers and descendants contain features (AB, BC) in all pairs of fathers and descendants. Let this value be 2%. From these two values, we know that, among all these 4% pairs prefixing with AB, half of them are followed by BC.

To state it formally, we call the pair  $(X, Y)$  as a featureset-pair, where  $X$  and  $Y$  are two sets of features (featureset). In the pair,  $X$  or  $Y$  can be replaced by a wild character \*, which denotes any possible feature including null. So, any feature set, including null set, contains \*. The following definition shows how the transactions can be derived from the DAG to find F-D rules.

**Definition 1.** Let  $g$  denote a DAG. Then the transactions generated for finding F-D rules are  $T-I(g) = \{(L, R) \mid l \text{ in } V(g) \wedge r \text{ in } V(g) \wedge l \neq r \wedge \text{is-ancestor}(l, r) = \text{true} \wedge L = F(l) \wedge R = F(r)\}$ .

According to the above definition, a transaction is created for each pair of father node and descendant node, and the content is a featureset-pair, where the first comes from the father and the second the descendant. The support of  $(X, *)$  is defined as the percentage of transactions in  $T-I(g)$  where  $L$  contains  $X$  and  $R$  contains \*. Similarly, the support of  $(X, Y)$  is the percentage of transactions in  $T-I(g)$  where  $L$  contains  $X$  and  $R$  contains  $Y$ . Besides, we have a minimum support threshold, and all the featureset-pairs with supports no less than the threshold are called frequent featureset-pairs. Finally, the support of frequent  $(X, Y)$  is divided by the support of frequent  $(X, *)$  and

the value is called the confidence of  $X \rightarrow Y$ . If the confidence is greater than the minimum confidence, the F-D rule  $X \rightarrow Y$  holds.

The second kind of inheritance rule is concerned with the inheritance relationships between ancestors and descendants. Let us consider the following scenario. Suppose we already know that the ancestors of John, i.e., John as well as all of his ancestors, possess some features, say A and B. Then, we may be interested in finding how likely the descendants of John, excluding John, will have some other features, say B and C. In the rule form, it can be expressed as  $AB \Rightarrow BC$ . For reference, we call this kind of rules as ancestors-descendants rule (A-D rule), because it analyzes how the features of ancestors affect those of the descendants.

Intuitively, the rule  $AB \Rightarrow BC$  can be determined as follows. First, we must find the percentage of persons whose ancestors contain features AB. Next, we must find the percentage of persons whose ancestors contain AB and whose descendants contain BC. From them, the rule can be determined.

To generate transactions for A-D rules, we select a set of persons as pivots, from each of which a transaction is generated. In other words, the number of pivots determines the number of transactions. However, it is not proper to include all persons as pivots, because some persons in DAG may only have few ancestors or few descendants. Using them as counting pivots may distort the mining result. For this reason, we allow the users to filter the data by setting the parameters *ugap* and *lgap* in function  $T(g, ugap, lgap)$ . The following definition states how the transactions can be generated from DAG to find A-D rules.

**Definition 2.** Let  $g$  denote a DAG. Then the transactions generated for finding A-D rules are  $T-3(g) = \{(L, R) \mid p \in T(g, ugap, lgap) \wedge L = F(ancestors(p) \cup \{p\}) \wedge R = F(descendants(p))\}$ .

In the above definition, a transaction is created for every person  $p$  in  $T(g, ugap, lgap)$ , and the content is a featureset-pair, where the first comes from the ancestors and the second from the descendants. After that, all the other measures can be defined as similar to the first inheritance rule; thus, they are omitted.

### 3 The Solution Method and the Generation of Transactions

According to the discussions in Section 2, the steps for finding inheritance rules can be summarized as follows.

1. Generate transactions from DAG data.
2. Find the supports of all frequent patterns ( $X, *$ ).
3. Find the supports of all frequent patterns ( $X, Y$ ).
4. Generate the inheritance rules from all frequent patterns ( $X, *$ ) and ( $X, Y$ ).

After step 1, we will have a set of generated transactions, denoted as  $t_1, t_2, \dots, t_z$ . Here, transaction  $t_i$  is formed of a pair  $(s1_i, s2_i)$ , where  $s1_i$  and  $s2_i$  are sets of items. For ease of reference, let us call the set containing  $s1_1, s1_2, \dots, s1_z$  as the antecedent transaction set, while the set containing  $s2_1, s2_2, \dots, s2_z$  the consequent transaction set. Before executing steps 2 and steps 3, let us label all the items in the two sets with different tags. Say, an item A appearing in the antecedent set may be relabeled as  $A^1$ ,

but the same item  $A$  in the consequent set should be relabeled as  $A^2$ . By this way, no items will appear in both sets, or stated in another way, all the items in the antecedent set are different from those in the consequent set.

After re-labeling items' name, we can regard each transaction  $t_i$  as a set of items, where each item in  $t_i$  either comes from  $s1_i$  or from  $s2_i$ . When an item is labeled with superscript "1", it is from  $s1_i$ . Otherwise, it is from  $s2_i$ . In other words, the current representation of transaction  $t_i$  allows us to clearly indicate where an item in  $t_i$  comes from, either from  $s1_i$  or from  $s2_i$ . Since the current transaction format has no difference from that of a typical transaction database, we can use the traditional association algorithm to mine this new database [2, 3, 4]. After mining, all the frequent itemsets can be represented as  $\{(X, Y)\}$ , where  $X$  is a set of items in  $s1_i$  and  $Y$  is a set of items in  $s2_i$ . Besides, since a subset of frequent itemset must be also frequent, if we found  $(X, Y)$  frequent, we will also have  $(X, *)$  frequent. Therefore, from the frequent itemsets, we can find the supports of all frequent patterns  $(X, *)$  as well as the supports of all frequent patterns  $(X, Y)$ . The final step is to generate all the inheritance rules from all frequent patterns  $(X, *)$  and  $(X, Y)$ . Most papers did not discuss this issue, for it can be easily done. In our case, it is still simple. We first sort all the frequent patterns  $(X, *)$ . Then we sort all frequent patterns  $(X, Y)$ . Finally, merging these two sources along with confidence computation generate all inheritance rules. Since existing methods can be used to solve steps 2 and 3 and since step 4 can be easily done, we will devote the remainder of this section to study how to generate transactions from DAG data.

### 3.1 Generate Transactions for F-D Rules

Let  $n$  and  $m$  denote the number of nodes and arcs in DAG  $g$ , respectively. Since each person has at most two parents, we have  $m \leq 2n$ . According to Definition 1, transactions are created for all pairs of nodes  $v_i$  and  $v_j$ , where  $v_i$  is an ancestor of  $v_j$ . Thus, we can do it by following algorithm GT-FD-1 in Fig. 3. In this algorithm, all the pairs of nodes are examined; thus, we have to examine  $O(n^2)$  pairs. For each pair of nodes, finding a path between them can be done by depth first search, which needs time  $O(m)=O(n)$ . Totally, we spend  $O(n^3)$  time to generate the transactions.

Although Algorithm GT-FD-1 is correct, it is inefficient because it needs  $n^2$  traversals. To reduce the redundancy, Algorithm GT-FD-2 in Fig. 4 does the traversal beginning from every node  $v_i$ . For all the nodes  $v_j$  encountered, it is certain that node  $v_i$  is an ancestor of node  $v_j$ ; thus, we can create a transaction for them. This way, we execute the traversal  $n$  times. Since each traversal can be done in time  $O(n)$ , the total time complexity is  $O(n^2)$ .

```

For  $i= 1$  to  $n$  do
  For  $j= 1$  to  $n$  do
    If node  $v_i$  has a path leading to node  $v_j$ 
      then create a transaction for the pair of  $v_i$  and  $v_j$ .

```

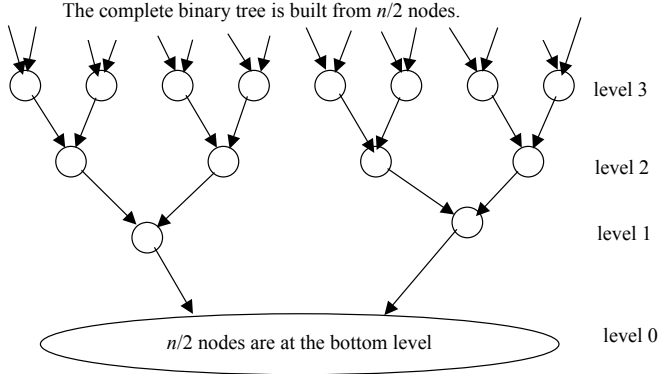
**Fig. 3.** Algorithm GT-FD-1

```

For  $i=1$  to  $n$  do
  Do a traversal beginning from node  $v_i$ 
  For every node  $v_j$  encountered
    Create a transaction for the pair of  $v_i$  and  $v_j$ .

```

**Fig. 4.** Algorithm GT-FD-2



**Fig. 5.** There are  $O(n^2)$  pairs of ancestors and descendants

**Theorem 1.** The time complexity of Algorithm GT-FD-2 is optimum, meaning that no algorithm can generate the transactions with a better time complexity.

**Proof.** Let us use a DAG shown in Fig. 5 to prove. In the figure,  $n/2$  nodes are used to build an upside-down binary tree, and the other  $n/2$  nodes are put at level 0. Note that the nodes at the top level are not necessarily at the same height. Every node in the tree has only one child and at most two parents, but the nodes at level 1 will have all the nodes at level 0 as their children. Obviously, all  $n/2$  nodes at level 0 have  $n/2$  ancestors in the tree. Thus, there are totally  $O(n^2/4)=O(n^2)$  pairs of ancestors and descendants. Since all these pairs need to be output, it is impossible to have an algorithm with time complexity less than  $O(n^2)$ .  $\square$

### 3.2 Generate Transactions for A-D Rules

Without loss of generality, assume that each node  $v_i$  is attached with a Boolean array  $F_{v_i}$  of  $C$  elements, where  $F_{v_i}(j)=1$  means feature  $j$  appearing in the node and 0 otherwise. Further, we assume that arrays  $S$  and  $D$  have the same structure as that of  $F_{v_i}$ . According to Definition 2, a transaction is created for every node  $v_i$  in  $T(g, ugap, lgap)$ , and the content of a transaction is a pair of featureset, where the first is the features of  $ancestors(v_i) \cup \{v_i\}$  and the second is the features of  $descendants(v_i)$ . So, we can generate the transactions by following Algorithm GT-AD-1 in Fig. 6. In this algorithm, every node will be set as pivot at most once. Let  $v_i$  denote the pivot. Then, we will do an upward search to find the features of all the ancestors and put them into  $S$ . After that, we will do a downward search to find the features of all the descendants

and put them into  $D$ . Finally, we create a transaction by combining  $S$  and  $D$  into a sequence. Here, the upward search and the downward search can be done by applying the breadth-first search algorithm; thus, they will examine  $O(n)$  arcs and  $O(n)$  nodes. For each node  $v_j$  that is currently visited, we need to unite the features in  $S$  or  $D$  with those in  $F_{v_j}$ , and it can be done in time  $O(C)$  for they are Boolean arrays of  $C$  elements. Putting them together, the time for processing each pivot node is  $O(nC)$ . Since we have at most  $n$  pivot nodes, the total time complexity is  $O(n^2C)$ .

In the following, we will further improve the time complexity of Algorithm GT-AD-1. To this end, we first do a topological search on DAG, and arrange all the nodes according to their topological order [14]. Without loss of generality, let  $v_1, v_2, \dots, v_n$  denote all the nodes arranged according to the topological order.

In Algorithm GT-AD-2, each node  $v_i$  is attached with three Boolean arrays  $F_{v_i}$ ,  $S_{v_i}$  and  $D_{v_i}$  of  $C$  elements, where  $F_{v_i}$  is used to store  $F(v_i)$ ,  $S_{v_i}$  to store  $F(\text{ancestors}(v_i) \cup \{v_i\})$  and  $D_{v_i}$  to store  $F(\text{descendants}(v_i))$ . Basically, the algorithm contains four loops. The first loop is to set the initial values of  $S_{v_i}$  and  $D_{v_i}$ . Since  $S_{v_i}$  includes the features of node  $v_i$ , the initial value of  $S_{v_i}$  is set as  $F_{v_i}$ . On the contrary, since  $D_{v_i}$  excludes the features of node  $v_i$ , the initial value of  $D_{v_i}$  is set as a zero array. The second loop is to pass down the features of ancestors to descendants. So, we iterate over the nodes  $v_1, v_2, \dots, v_n$  to pass down the features and store the features inherited along with those originally existed in node  $v_j$  into  $S_{v_j}$ . The third loop is to send the features of descendants up to ancestors. So, we iterate over the nodes  $v_n, v_{n-1}, \dots, v_1$  to repeatedly send the features up and store the features coming from the descendants into  $D_{v_j}$ . Finally, the fourth loop will output all feasible pairs of  $(S_{v_i}, D_{v_i})$  as transactions.

Fig. 8 shows a DAG with 8 nodes, where the number beside each node is its topological order and the Boolean array in a node denotes  $F_{v_i}$ . In the second loop, we examine the nodes in accordance with the topological order. In each examination, we will pass down the features to its children. After that, we will go to the third loop, where we examine the nodes in an order exactly contrary to the topological order. That is, the examination sequence is nodes 8, 7, 6, 5, 4, 3, 2 and finally 1. In each examination, we will send the features up to its parents. After the third loop, our result is shown in Fig. 9, where the Boolean array in each node is  $D_{v_i}$ . Suppose only nodes 3, 4 and 5 are our transaction nodes, then the generated transaction would be  $((1,0,1,0,0), (0,1,0,0,0)), ((1,1,0,1,0), (1,0,1,0,1))$  and  $((1,1,0,1,0), (1,1,1,0,1))$ .

```

For  $i = 1$  to  $n$  do
  If  $v_i$  is not in  $T(g, \text{ugap}, \text{lgap})$  then exit.
   $S = \emptyset, D = \emptyset$ .
  For every node  $v_j$  in  $\text{ancestors}(v_i) \cup \{v_i\}$  do  $S = S \cup F_{v_j}$ .
  For every node  $v_j$  in  $\text{descendants}(v_i)$  do  $D = D \cup F_{v_j}$ .
  Create a transaction with  $(S, D)$ 

```

**Fig. 6.** Algorithm GT-AD-1

```

Do a topological search to arrange the nodes
For  $i = 1$  to  $n$  do  $S_{v_i} = F_{v_i}$  and  $D_{v_i} = \text{zero array}$ ;
For  $i = 1$  to  $n$  do
    For every arc  $(v_j, v_i)$  do  $S_{v_j} = S_{v_j} \cup S_{v_i}$ ;
For  $i = n$  to  $1$  do
    For every arc  $(v_j, v_i)$  do  $D_{v_j} = D_{v_j} \cup D_{v_i} \cup F_{v_i}$ ;
For  $i = 1$  to  $n$  do
    If  $v_i$  is not in  $T(g, \text{ugap}, \text{lgap})$  then exit.
    Create a transaction with  $(S_{v_i}, D_{v_i})$ 

```

Fig. 7. Algorithm GT-AD-2

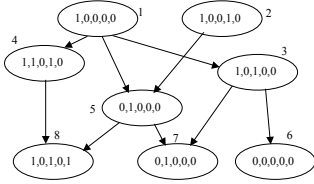
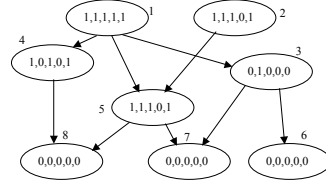


Fig. 8. A DAG with 8 nodes

Fig. 9. Compute  $D_{v_i}$ 

Now, we will analyze the time complexity of Algorithm GT-AD-2, which contains a topological search and four loops. At first, the topological search can be done in time  $O(n)$  for the DAG with  $O(n)$  arcs and  $n$  nodes. The first loop and the fourth loop can both be done in time  $O(nC)$ , because they examine every node once and in each examination they need to set an array of  $C$  elements or output a transaction of  $2C$  elements. The second loop and the third loop can both be done in time  $O(nC)$ , because they examine every arc once and in each examination they need one or two union operations between arrays of  $C$  elements. In sum, the total time required is  $O(nC)$ .

**Theorem 2.** The time complexity of Algorithm GT-AD-2 is optimum, meaning that this time complexity is impossible to be further improved.

**Proof.** We have  $O(n)$  transaction nodes in the DAG and each transaction contains two feature arrays of  $C$  elements. To output all these transactions we need time at least  $\Omega(nC)$ , not to mention the time to generate the transactions. So, we conclude that Algorithm GT-AD-2 is the best time complexity algorithm.

## 4 Conclusion

The problem studied is to find two kinds of inheritance rules from DAG data. To mine these rules, an algorithm consisting of four stages is developed. This paper has some possible extensions. First, as mentioned in the introduction the data in many applications can be represented by using DAG data structure. When more applica-

tions about DAG data are considered, we believe that many other kinds of association rules can be developed. Second, since DAG data is just a specialization of graph data, it is a reasonable extension that we may consider how to define association rules in graph data. Finally, we can associate a quantitative value with each arc or node in the DAG to denote some kind of attributes meaningful in practical situations. And the problem becomes how to find quantitative inheritance rules from DAG data.

## Acknowledgements

This research was supported in part by the MOE Program for Promoting Academic Excellence of Universities under the grant number 91-H-FA07-1-4.

## References

1. Chen, M.-S., Han J., Yu, P.S.: Data mining: an overview from a database perspective. *IEEE Transactions on Knowledge and Data Engineering* 8 (1996) 866-883
2. Agrawal, R., Imielinski, T., Swami, A.: Mining association rules between sets of items in large databases. In: *Proceedings of the ACM SIGMOD International Conference on Management of Data*. (1993) 207-216
3. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules. In: *Proceedings of the 20th VLDB Conference*. (1994) 478-499
4. Liu, J., Pan, Y., Wang, K., Han, J.: Mining frequent item sets by opportunistic projection. In: *Proceedings of the 2002 Int. Conf. on Knowledge Discovery in Databases*. (2002)
5. Kuok, C. M., Fu, A. W., Wong, M. H.: Mining fuzzy association rules in databases. *SIGMOD Record* 27 (1998) 41-46
6. Clementini, E., Felice, P.D., Koperski, K.: Mining multiple-level spatial association rules for objects with a broad boundary. *Data and Knowledge Engineering* 34 (2000) 251-270
7. Wijzen, J., Meersman, R.: On the complexity of mining quantitative association rules. *Data Mining and Knowledge Discovery* 2 (1998) 263-281
8. Koperski, K., Han, J.: Discovery of spatial association rules in geographic information databases. In: *Proc. 4th International Symposium on Large Spatial Databases (SSD95)*. (1995) 47-66
9. Lu, H., Feng, L., Han, J.: Beyond intra-transaction association analysis: mining multi-dimensional inter-transaction association rules. *ACM Transactions on Information Systems* 18 (2000) 423-454
10. Freitas, A.A.: On rule interestingness measures. *Knowledge-Based Systems* 12 (1999) 309-315
11. Lee, C. H., Lin, C. R., Chen, M. S.: On mining general temporal association rules in a publication database. In: *Proceedings of the 2001 IEEE International Conference on Data Mining*. (2001) 337-344
12. Zhuge, H.: Inheritance rules for flexible model retrieval. *Decision Supports Systems* 22 (1998) 379-390
13. Han, J., Kamber, M.: *Data Mining*, Morgan Kaufmann, San Francisco. (2001)
14. Shaffer, C. A.: *A Practical Introduction to Data Structures and Algorithm Analysis: JAVA edition*, Prentice Hall Inc. (1998)

# Mining DAG Patterns from DAG Databases

Yen-Liang Chen<sup>1</sup>, Hung-Pin Kao<sup>1</sup>, and Ming-Tat Ko<sup>2</sup>

<sup>1</sup> Dept. of Information Management, National Central Univ, Chung-Li, Taiwan 320  
{ylchen,kris}@mgt.ncu.edu.tw

<sup>2</sup> Institute of Information Science, Academia Sinica, Taipei, Taiwan 115  
mtko@iis.sinica.edu.tw

**Abstract.** Data mining extracts implicit, previously unknown and potentially useful information from databases. Many approaches have been proposed to extract information, and one of the most important ones is finding frequent patterns in databases. Although much work has been done to this problem, to the best of our knowledge, no previous research studies how to find frequent DAG (directed acyclic graph) patterns from DAG data. Without such a mining method, the knowledge cannot be discovered from the databases storing DAG data such as family genealogy profiles, product structures, XML documents and course structures. Therefore, a solution method containing four stages is proposed in this paper to discover frequent DAG patterns from DAG databases.

## 1 Introduction

Data mining extracts implicit, previously unknown and potentially useful information or patterns from databases. Many approaches have been proposed to extract information. One of the most important ones is finding frequently occurred sub-structures, called frequent patterns, from databases. In the past, much work has been done to this problem. Various variants of frequent patterns have been proposed. Examples of these variants include finding frequent itemsets from transaction databases, finding frequent sequential patterns from sequence data, finding frequent traversal patterns from web traversal data and finding frequent sub-trees or sub-graphs from databases containing tree data or graph data. In the following, we give a more detailed discussion about the last problem.

The last problem aims to find tree patterns [1, 2, 3] or graph patterns [4, 5, 6, 7, 8] from databases containing graph data or tree data. In fact, these two approaches are closely related, since graphs are a generalization of trees, and trees are a specialization of graphs. Therefore, if we use the algorithms for mining graph patterns to mine the database of tree data, we can still find tree patterns, though we may spend a longer time to do so. On the other hand, the algorithms for mining tree patterns are not able to discover graph patterns from graph data, because of the limitation imposed by the algorithms. This comparison shows their relative strength and weakness is that, though the algorithms for mining tree patterns are more limited in their applicability, they are more efficient. On the contrary, though the algorithms for mining graph patterns may spend a much longer time, the problems they can solve are more general. These opposite characteristics indicate how these two approaches can be in co-



existence with each other. When the problem on hand is still under the scope of tree pattern mining, we prefer using the tree mining algorithms to do the job, for this saves run time. On the other hand, when the problem on hand goes beyond the scope of tree pattern mining, we may resort to the graph pattern mining algorithms, though they need a much longer time to run.

Although a lot of works have been done on mining tree patterns or graph patterns, to the best of our knowledge, no existing research has ever considered the problem of mining DAG (directed acyclic graph) patterns from a DAG database. In many applications, the underlying data are of DAG structure. For example, family genealogy profiles, web navigation patterns, bills of materials in production control, XML documents, task precedence relations and course prerequisite structures can all be represented as DAG data. When we are given a database of DAG data, the traditional tree pattern mining methods would be unable to find the patterns from them, because the data structure is no longer tree but now becomes acyclic-graph structured. On the other hand, though we can use the algorithms for mining graph pattern to do the job, the spending time would be terribly long. In view of these difficulties, a requirement arising immediately is to have efficient algorithms that can mine DAG patterns from DAG databases.

The paper is organized as follows. In Section 2, we formally define the problem. Next, the solution method is given in Section 3. In Section 4, experiments are performed to show the performance of the proposed algorithm. Finally, the conclusion is drawn in Section 5.

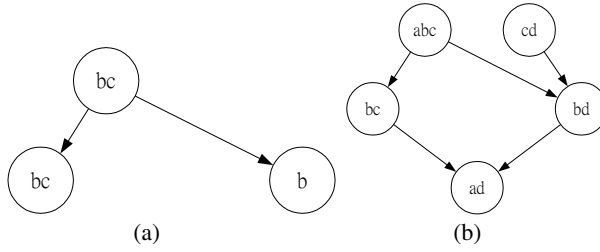
## 2 The Problem

Let  $D = (V, E)$  denote a directed acyclic graph, where  $V = \{v_1, v_2, \dots, v_n\}$  is a set of  $n$  nodes and  $E = \{e_1, e_2, \dots, e_m\}$  is a set of  $m$  arcs. Let  $FV = \{fv_1, fv_2, \dots, fv_p\}$  denote the set of all the features in the node set. Each node  $v_x$  is associated with a set  $F_{v_x}$  of features, where  $F_{v_x} \subset FV$ .

For ease of presentation, the following notations and functions are defined before getting into the detailed discussion.

1.  $V(g)$  is the set of nodes in DAG  $g$ .
2.  $E(g)$  is the set of arcs in DAG  $g$ .
3.  $F(v_x)$  is the features associated with node  $v_x$ .
4.  $head(e_y) = v_1$  and  $tail(e_y) = v_2$  if  $e_y = (v_1, v_2)$ .
5.  $|E(g)| = m$ , if there are  $m$  arcs in DAG  $g$ .
6.  $|V(g)| = n$ , if there are  $n$  nodes in DAG  $g$ .
7.  $A(g) = \{v \mid v \in V(g) \wedge (\forall e, e \in E(g) \wedge tail(e) \neq v)\}$

Here,  $A(g)$  denotes the set of nodes which have no ancestors in DAG  $g$  and  $D(g)$  the set of nodes which have no descendants in DAG  $g$ . For specificity, we call the nodes in  $A(g)$  as the source nodes and the nodes in  $D(g)$  as the sink nodes. If we apply the function  $A(g)$  to the DAG in Fig. 1(b), we will have two nodes, where one is attached with features  $\{a, b, c\}$  and the other  $\{c, d\}$ .



**Fig. 1.** (a) DAG  $g_1$ . (b) DAG  $g_2$

In the following, we give two definitions concerning DAG, which will be used to define what a DAG pattern is.

**Definition 1.**  $g_s$  is a subgraph of  $g$  if the following conditions are satisfied. (1)  $g_s$  is a DAG, (2)  $V(g_s) \subset V(g)$ , and (3)  $E(g_s) \subset E(g)$ . Accordingly, we have the function  $subgraphof(g_s, g)$ .

Another important relation between two DAG data is “contained-by”, which means the features of a graph is contained by another graph of the same size and we use  $\subset$  to denote this operation.

**Definition 2.** For two graphs  $g_1, g_2$  with the same number of nodes, we say  $g_1 \subset g_2$ , if the following two conditions are satisfied. (1) For every node  $u$  in  $g_1$ , we can map it to a node  $\theta(u)$  in  $g_2$  in one-to-one correspondence such that  $F(u) \subset F(\theta(u))$ . (2) For every arc  $e_1=(u, v)$  in  $g_1$ , there exists an arc  $e_2$  in  $g_2$  such that  $e_2 = (\theta(u), \theta(v))$ .

Next, we define the function  $instanceof(p, g)$  to determine if a DAG pattern  $p$  is an instance of a DAG  $g$ . It will return true if  $\exists i, subgraphof(i, g) \wedge p \subset i$ . This means that,  $p$  is an instance of  $g$  if we can find a subgraph of  $g$ , say  $i$ , so that  $i$  contains  $p$ . For example, please refer to Fig. 1. It is easy to see that  $instanceof(g_1, g_2)$  is true because we can find a subgraph  $i$  in  $g_2$  such that  $g_1 \subset i$ .

Having defined the function  $instanceof(p, g)$ , we now define the support of a pattern  $p$ . Let  $DAGS$  be the set of DAGs that we are about to mine. Then, the support of pattern  $p$  in  $DAGS$  can be determined as follows.

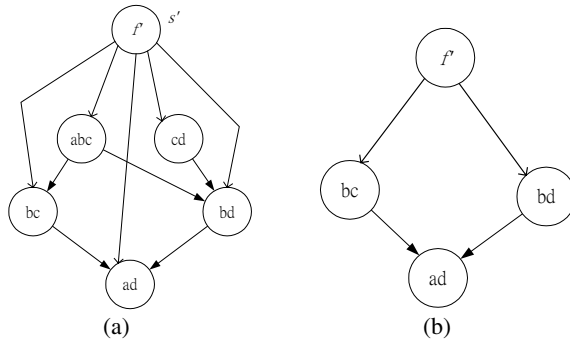
$$sup(p, DAGS) = \frac{|\{i \mid i \in DAGS \wedge instanceof(p, i)\}|}{|DAGS|}$$

Finally, a DAG pattern  $p$  is a frequent pattern if its support exceeds the minimum support threshold specified by the users.

### 3 The Algorithm

This section focuses on finding a special type of DAG patterns, called the pyramid pattern, which is a DAG pattern but with only one source node. Formally speaking, a DAG pattern  $p$  is a pyramid pattern if  $|A(p)|=1$ . For example, Fig. 1(a) is a pyramid pattern. Focusing on finding pyramid patterns can simplify the design of the algorithm. However, this does not harm our ability to find DAG patterns hidden in the

original DAG database, because by slightly transforming the original DAG data all DAG patterns can still be found from the new DAG database. The following are the steps to do this data transformation. First, we create a virtual source node  $s'$  with a special feature  $f'$ , where  $f' \notin FV$ . Then, we create arc  $(s', u)$  for each node  $u$  in each DAG data. For example, the DAG data  $g_2$  in Fig. 1(b) can be transformed to the DAG data  $g_2'$  shown in Fig. 2(a). Let  $DAGS'$  denote the set of DAG data obtained after this transformation. In  $DAGS'$ , the algorithm would find all pyramid patterns satisfying the properties that (1) the source node of a pyramid pattern must contain feature  $f'$  and (2) the arcs leaving from the source node must enter the nodes at the second level. Fig 2(b) shows a pyramid pattern that we can find from the extended DAG data in Fig. 2(a).



**Fig. 2.** (a) DAG  $g_2'$ . (b) A pyramid pattern in  $g_2'$

The following theorem shows the transformation is correct.

**Theorem 1.** There is a one-to-one correspondence between DAG patterns in  $DAGS$  and pyramid patterns in  $DAGS'$ .

Our algorithm contains four phases, and we will introduce it one by one in the following subsections.

### 3.1 Find the Set of Frequent Feature Sets

There are two reasons for finding frequent feature sets ( $FFS$ ). First, we can remove the features associated with a node in a DAG which are not frequent. Finding  $FFS$  earlier can remove the unnecessary feature data in the initial stage of the algorithm. Second, knowing  $FFS$  can reduce the number of candidate patterns generated in later stages, and this improves the efficiency of the algorithm.

The following theorem shows why  $FFS$  can reduce the number of candidate patterns.

**Theorem 2.** Let  $u$  denote a node in a frequent pattern  $fp$ . Then the features associated with node  $u$  must be a feature set in  $FFS$ .

According to this property, when generating candidate patterns, we only need to consider the feature sets in  $FFS$ , and other feature sets can be excluded from consid-

eration. It is obvious that *FFS* can be found by applying the well-known methods for mining frequent itemsets [9, 10]; so, we omit the details.

### 3.2 Remove the Unnecessary Feature Data

Having obtained the *FFS*, we then remove the unnecessary feature data from the input. To do so, we scan the features of every node in each DAG. If we find any feature that does not appear in *FFS*, it can be removed. For example, assume  $FFS = \{\{a\}, \{b\}, \{e\}, \{h\}, \{b, e\}, \{a, b\}\}$ . Then, all the features not in the set  $\{a, b, e, h\}$  are removed.

### 3.3 Find Linear Patterns

The steps in this and the next sections are based on the property that a pattern would not be frequent unless all of its sub-patterns are frequent, meaning that we can construct candidate patterns from their frequent sub-patterns. Therefore, we first find linear patterns, because linear pattern is not only a special pyramid pattern but also a sub-pattern of general pyramid patterns. To obtain all linear patterns, we begin from smaller linear patterns and gradually expand them to larger linear patterns by applying this property once again, i.e., a smaller frequent linear pattern is a sub-pattern of a larger frequent linear pattern.

To each node in the pattern, two data are attached, where the first is its feature set and the second is its depth. The algorithm iterates phase by phase. First, we construct a linear pattern of a single node with feature  $f'$ . Afterwards, the frequent linear patterns found in the previous phase are used to construct the candidate set of linear patterns in the next phase. After executing the  $k$ -th phase, we will find all frequent linear patterns with  $k$  nodes. Finally, all frequent linear patterns are stored in the variable *FreqLinear*.

### 3.4 Find the Pyramid Patterns

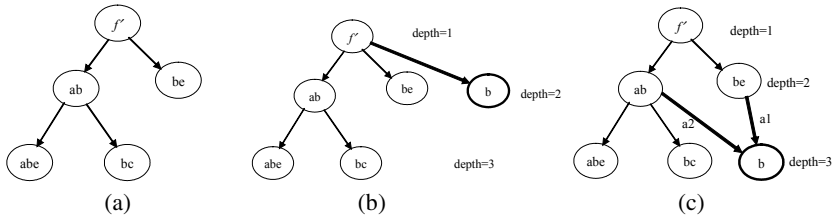
With each node on a pyramid pattern, two data are associated, where the first is its feature set and the second is the depth of this node in the pattern, where the depth of a node is defined as follows. (1)  $depth(root)=1$ , and (2)  $depth(v)=\max\{depth(u)|arc(u, v) \text{ exists in the pattern}\}+1$ .

Suppose we are given frequent pyramid patterns of depth  $i$  and with  $j-1$  nodes. Then, how can we generate the candidate set of pyramid patterns of depth  $i$  but with  $j$  nodes? To this end, we need to add one new node, say  $v$ , into an original pattern without increasing its total depth. But, how can we connect the new node  $v$  to the other nodes without violating the pyramid property? The answer to this question depends on the depth of node  $v$  where we will put it. Obviously, node  $v$  cannot be at depth 1, for this will make the pattern contain more than one source. Suppose we want to put node  $v$  at depth  $k$ , where  $2 \leq k \leq i$ . To ensure node  $v$  staying at depth  $k$ , the following conditions should be satisfied.

1. We must create arc  $(u, v)$  for at least one node  $u$  at depth  $k-1$ .
2. It doesn't matter if arc  $(u, v)$  is added or not, if the depth of node  $u$  is from 2 to  $k-2$ .
3. It is not allowed to have arc  $(u, v)$ , if the depth of node  $u$  is no less than  $k$ .

With the above rules, the constructed pattern will have at least one arc connecting a node at depth  $k-1$  to node  $v$ , and all the other arcs connecting to node  $v$  are from those nodes whose depths are smaller than  $k$ . So, the newly constructed pattern is still a pyramid pattern and the newly added node stays at depth  $k$ . By iterating the value of  $k$  from 2 to  $i$ , we will finally generate all pyramid patterns of depth  $i$  and with  $j$  nodes.

For example, let us consider the original pattern shown in Fig. 3(a), which has 5 nodes and has depth 3. To expand this pattern with one more node but without increasing the depth, the new node can be added either at depth 2 or at depth 3. Suppose we want to add a new node with feature  $\{b\}$ . When it is put at depth 2, it will be like the one shown in Fig. 3(b). But, if we put the new node at depth 3, there are totally three possible combinations as shown in Fig. 3(c).



**Fig. 3.** Add a new node into a pattern with five nodes and depth 3

```

Function Expand(Pyramid pp){
  let Ret be a null set.
  for each fs in FFS {
    for depth  $d = 2$  to  $i$  {
      create a new node  $v$  with features fs.
      let  $S_{d-1}$  be the set of nodes of depth  $d-1$ .
      let  $S_{<d-1}$  be the set of nodes of depth from 2 to  $d-2$ .
      for all nonempty subsets  $v1$  in  $S_{d-1}$ {
        for all subsets  $v2$  in  $S_{<d-1}$ {
          make a new pattern npp by adding edges
            from all nodes in  $v2$  and  $v1$  to node  $v$ .
          if npp is not a duplicate pattern then add npp to Ret.
        }
      }
    }
  }
  return Ret.
}

```

**Fig. 4.** The pattern expansion algorithm

Fig. 4 shows how the expansion can be done. For a given input pattern  $pp$ , the function obtains a set  $Ret$  of extended patterns, where each extended pattern has one

more node than  $pp$  but has the same depth as  $pp$ . However, the generated pattern should not be isomorphic to another candidate pattern produced before. If so, then this pattern is a duplicate pattern and should be deleted before it is put into the set *Ret*. All these generated patterns are the candidates of frequent patterns.

Let us define  $PP_{ij}$  as the set of frequent pyramid patterns of depth  $i$  and with  $j$  nodes. For example, the pattern of Fig. 3(a) is in  $PP_{3,5}$ , and the patterns of Fig 3(b) and Fig. 3(c) in  $PP_{3,6}$ . According to the definition of  $PP_{ij}$ , it is clear that they satisfy the following properties: (1)  $j \geq i$ , and (2) Since  $PP_{k,k}$  is the set of frequent linear patterns of  $k$  nodes, all the sets like  $PP_{2,2}$ ,  $PP_{3,3}$ , ... can be obtained directly from *FreqLinear*, which was obtained in the preceding phase.

The main program, *MinePP()*, is shown in Fig. 5, which contains two loops. The central part of the program is to find  $PP_{ij}$ . If it succeeds, the next cycle of the inner loop goes to find  $PP_{i,j+1}$ . Otherwise, the inner loop stops and the next cycle of the outer loop goes to find  $PP_{i+1,i+1}$ . This will be repeatedly executed until we cannot find any frequent set. In short, the algorithm finds all  $PP_{ij}$  in the following order:  $PP_{2,2} \rightarrow PP_{2,3} \rightarrow PP_{2,4} \rightarrow \dots$ ;  $PP_{3,3} \rightarrow PP_{3,4} \rightarrow PP_{3,5} \rightarrow \dots$ ;  $PP_{4,4} \rightarrow PP_{4,5} \rightarrow PP_{4,6} \rightarrow \dots$

To successfully execute the above procedure, two problems should be solved: (1) We must know how to compute  $PP_{k,k}$  for  $2 \leq k$ , and (2) When we are given  $PP_{i,j-1}$ , we must know how to compute  $PP_{ij}$  from  $PP_{i,j-1}$ . The first problem is easy to answer, since it can be obtained from *FreqLinear*. Therefore, the only problem is how to compute  $PP_{ij}$  from  $PP_{i,j-1}$ . The following explains how it can be done.

1. For every pattern in  $PP_{i,j-1}$ , using the expansion program in Fig. 4 to add a new node into it. Let us denote the resulting set as  $PP_{ij}(1)$ .
2. Remove from  $PP_{ij}(1)$  those patterns which have infrequent sub-patterns. Let us denote the remaining set as  $PP_{ij}(2)$ .
3. Compute the supports of the patterns in  $PP_{ij}(2)$ . Remove the infrequent patterns.

The resulting set is  $PP_{ij}$ .

Step 2 checks if a pattern has any infrequent sub-patterns. If so, then the pattern will be removed from  $PP_{ij}(1)$ . However, we do not need to check all possible sub-patterns, because except the newly added node all the other nodes are existing nodes and the sub-patterns formed of these existing nodes are known frequent. Thus, we only need to check if the sub-patterns containing the newly added node are frequent or not. To do this check, a better way is to enumerate all the paths from the root node to the new node. Note that all these paths are linear sub-patterns. Further, they must be frequent; otherwise, the whole pattern is not frequent either. Therefore, we only check if these linear sub-patterns are in *FreqLinear* or not. If any of them do not, the whole pattern is infrequent and can be discarded. For example, all the paths from the root node to the new node in Fig. 3(c) are ( $\{f\}$ ,  $\{be\}$ ,  $\{b\}$ ) and ( $\{f\}$ ,  $\{ab\}$ ,  $\{b\}$ ). If any of these linear patterns are not frequent, then the candidate pattern of Fig. 3(c) can be removed.

```

Function MinePP() {
    let FreqPyramidPattern be a null set.
    for(int i=2; ;i++){
        for(int j=i; ;j++){
            compute PP[i,j] from PP[i,j-1],
            if PP[i,j] is empty {
                break.
            } else {
                add PP[i,j] to FreqPyramidPattern.
            }
        }
        if PP[i,j] is empty and PP[i+1,i+1] is empty {
            break;
        }
    }
}

```

**Fig. 5.** The main program

## 4 Performance

To study the performance, the algorithm is implemented in Java language and tested on a Pentium III-933 Windows-2000 system with 768 megabytes of main memory. Besides, we use JVM (J2RE 1.3.0 IBM build cn130-20010207) as the Java execution environment. We generate the synthetic data by modifying the well-known synthetic data generation program [9, 10].

The first parameter  $|FV|$  is the total number of features. The second parameter  $|F_{v_x}|$  is used to determine the number of features attached to a node. To determine its size, we picked up a value from a Poisson distribution with mean  $|F_{v_x}|$ . The third parameter  $|D|$  helps us to determine the number of children of each node. Its value is determined by picking up a value from a Poisson distribution with mean  $|D|$ . In the experiment, we set these three parameters as 1000, 10 and 2, respectively.

To model the phenomenon that a child node is often somewhat similar to its parents, some fraction of features of a child node is inherited from its parents. The fourth parameter  $I$  is designed for this purpose, where the inheritance fraction of a node is determined by picking up a value from a Normal distribution with mean  $I$  and standard deviation 0.1. When the value of  $I$  is larger, a child node will be more similar to its parents, and this will produce more frequent patterns. Our experiment sets  $I$  as either 0.6 or 0.8.

For a child node with  $|F_{v_x}|$  features and inheritance fraction  $I$ ,  $\lfloor |F_{v_x}| \times I \rfloor$  features are inherited from its parent, while the other features are selected from maximal potentially frequent feature sets. If the selected feature set is longer than the space left in the transaction, we sequentially pick up the features in the set until the predestined transaction size has reached. On the other hand, if the selected feature set is shorter, the remaining features are generated at random. The generation of maximal poten-

tially frequent feature sets  $L$  follows the traditional way of generating maximal potentially large itemsets [9, 10]. In our experiment, we set  $|L|=300$  and  $|f|=2$ .

In each run of simulation, we first generate a big DAG, containing more than 10000 nodes. From every node a DAG data is created by using it as the root and deleting the nodes and arcs not reachable from the root. In this way, the original DAG is partitioned into a lot of DAG data, as many as the total number of nodes. Finally, we use this set of DAG data as the input database. Three experiments are performed. The first and the second compare the algorithm's execution times for different numbers of nodes and for different support thresholds. Finally, the third compares the numbers of generated frequent feature sets (FFS) and frequent pyramid patterns (FPP) for different support thresholds.

In the first experiment, we set  $I=0.6$ ,  $|f|=2$  and the minimum support=2% (for simplicity, we abbreviate  $I=0.6$ ,  $|f|=2$  as I0.6f2.) To compare running time for different values of  $|V|$ , we generate the data sets for  $|V|=10000, 15000, 20000, 25000$  and 30000. Fig. 6 shows that the execution time increases in linear relation with the size of the network. In the second experiment, we set I0.6f2 and  $|V|=10000$ , and the minimum support value is varied from 1.75% to 2.75%. Fig. 7 shows that the execution time decreases sharply when the minimum support increases. In the third experiment, to compare the numbers of FFS and FPP for different support thresholds we set I0.6f2 and  $|V|=10000$  and vary the minimum support value from 1.75% to 2.75%. The results in Fig. 8 and Fig. 9 indicate that the smaller the minimum support is, the more frequent patterns we have. Basically, the results of all these three experiments match our expectation. However, comparing the result in Fig. 6 with that in Fig. 7, we find that changing the minimum support value has a stronger impact on the performance of the algorithm than changing  $|V|$ . Therefore, when the efficiency of the algorithm is not satisfactory, it would be easier to have an acceptable performance by changing the minimum support value than by reducing the problem size.

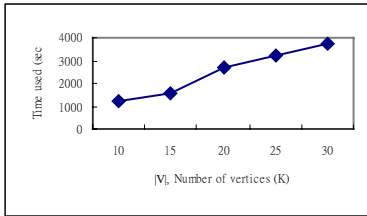


Fig. 6. Time vs.  $|V|$

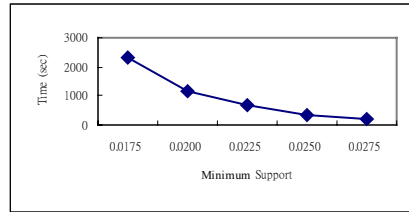


Fig. 7. Time vs. supports

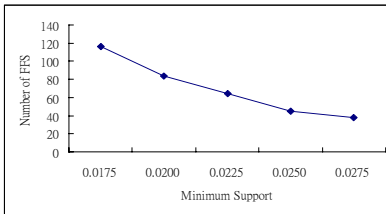


Fig. 8. FFS# vs. supports

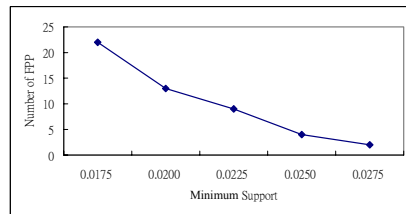


Fig. 9. FPP# vs. supports



## 5 Conclusions

The problem studied in this paper is to find DAG patterns from DAG data. In this paper, an algorithm consisting of four stages is developed, including (1) find the set of frequent feature sets, (2) remove the unnecessary feature data, (3) find the linear patterns and (4) find the pyramid patterns.

## Acknowledgments

The research was supported in part by the MOE Program for Promoting Academic Excellence of Universities under the Grant Number 91-H-FA07-1-4.

## References

1. Lin, X., Liu, C., Zhang, Y., Zhou, X.: Efficiently computing frequent tree-like topology patterns in a web environment. In: Proceedings of Technology of Object-Oriented Languages and Systems. (1999)
2. Wang, K., Liu, H.Q.: Discovering structural association of semistructured data. IEEE Trans. on Knowledge and Data Engineering **12** (2000) 353-371
3. Wang, K., Liu, H.Q.: Mining is-part-of association patterns from semistructured data. In: Proceedings of the 9th IFIP 2.6 Working Conference on Database Semantics. (2001)
4. Inokuchi, A., Washio, T., Motoda, H.: An Apriori-based Algorithm for mining frequent substructures from graph data. In: Proceeding of the 4<sup>th</sup> European Conference on Principles and Practices of Knowledge Discovery in Databases. (2000) 13-23
5. Kuramochi, M., Karypis, G.: Frequent subgraph discovery. In: Proceedings of the IEEE International Conference on Data Mining. (2001)
6. Nanopoulos, A., Manolopoulos, Y.: Mining patterns from graph traversals. Data and Knowledge Engineering **37** (2001) 243-266
7. Yan, X., Han, J.: gSpan: graph-based substructure pattern mining. In: Proceedings of 2002 Int. Conf. on Data Mining (ICDM'02), Maebashi, Japan. (2002)
8. Yan, X., Han, J.: CloseGraph: mining closed frequent graph patterns. In: Proceedings of 2003 ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining (KDD'03), Washington, D.C. (2003)
9. Agrawal, R., Imielinski, T., Swami, A.: Mining association rules between sets of items in large databases. In: Proceedings of the ACM SIGMOD International Conference on Management of Data. (1993) 207-216
10. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules. In: Proceedings of the 20th International Conference on Very Large Data Bases. (1994) 478-499

# Mining Class Outliers: Concepts, Algorithms and Applications

Zengyou He<sup>1</sup>, Joshua Zhexue Huang<sup>2</sup>, Xiaofei Xu<sup>1</sup>, and Shengchun Deng<sup>1</sup>

<sup>1</sup> Department of Computer Science and Engineering, Harbin Institute of Technology, China  
zengyouhe@yahoo.com, {xiaofei,dsc}@hit.edu.cn

<sup>2</sup> E-Business Technology Institute, The University of Hong Kong, Pokfulam, China  
jhuang@eti.hku.hk

**Abstract.** Detection of outliers is important in many applications and has attracted much attention in the data mining research community recently. However, most existing methods are designed for mining outliers from a single dataset without considering the class labels of data objects. In this paper, we consider the *class outlier detection problem*, i.e., “given a set of observations with class labels, find those that arouse suspicions, taking into account the class labels.” By generalizing two pioneering contributions in this field, we propose the notion of class outliers and practical solutions by extending existing outlier detection algorithms to detect class outliers. Furthermore, its potential applications in CRM (customer relationship management) are discussed. The experiments on real datasets have shown that our method can find interesting outliers and can be used in practice.

## 1 Introduction

A well-quoted definition of outliers given by Hawkins[3] states that “an outlier is an observation that deviates so much from other observations as to arouse suspicion that it was generated by a different mechanism”. The majority of the past research efforts have been focused on the problem of detecting deviants in a single dataset without considering the class labels of data objects.

**Problem 1 (Traditional Outlier Detection).** Given a set of objects, find those that deviate significantly from the rest, regardless of class labels.

As shown in [1,2], it does not cover the important practical situations where we have collections of points with class labels. Consider the following illustrative example of customer analysis in CRM:

**Example 1.** Customer segmentation is to divide the entire customer population into smaller groups, called *customer segments*. As time goes on, customers shift among segments. One can move from the loyal segment to the vulnerable segment in terms of profits. Therefore, it is critical to identify these potential losers. From the viewpoint of outlier detection, even though everything may look “normal” when we ignore segment types in the whole customer base, they may deviate with respect to the loyal segment and look “normal” with respect to the vulnerable segment.

Therefore, we consider the class outlier detection problem.

**Problem 2 (Class Outlier Detection).** Given a set of observations with class labels, find those that arouse suspicions, taking into account the class labels.

To our knowledge, the class outlier detection problem has only been explicitly considered in [1,2]. Unfortunately, the two papers only consider a special case of class outlier. In [1], the class outlier is defined as “semantic outlier”. A semantic outlier is a data point, which behaves differently with other data points in the same class, while looks ‘normal’ with respect to data points in another class. In contrast, the authors of [2] proposed the cross-outlier detection problem. That is, given two sets (or classes) of objects, find those that deviate with respect to the other set.

In this paper, we target at detecting outliers in the classified data. We define the class outliers and propose the algorithms to detect the class outliers. Furthermore, how to apply it in CRM is discussed and its effectiveness is evaluated on real datasets.

## 2 Definitions of Class Outliers

**Definition 1. (Class Outlier Detection).** Given a set of observations with class labels, find those that arouse suspicions, taking into account the class labels.

Firstly, we introduce two basic types of class outliers: *local class outlier* and *reference class outlier*. And some notions to be used are listed in Table 1.

**Table 1.** Notations

Notion	Meaning
DB	Set of objects $DB=\{C_1, C_2, \dots, C_k\}$ , $C_i \cap C_j = \emptyset$ and $C_1 \cup C_2 \cup \dots \cup C_k = DB$
$ DS $	The number of elements in set $DS$
$Sp(DB)$	We let the span of a set $DB$ , denoted by $Sp(DB)$ , be the set of all unions of subsets of $DB$ . That is, $Sp(DB) = \{ \bigcup_{C \in B} C \mid B \subseteq DB \}$
$C_i$	Set of objects with class label $cl_i$
$cl_i$	The class label for $C_i$
$T$	The target set for class outlier mining, $T \in DB$

**Definition 2. (Local Class Outlier Detection).** Given  $DB$  and a target set  $T$ , for an object  $p \in T$ , the local class outlier factor of  $p$ , denoted as  $LCOF(p)$ , captures the degree to which we call  $p$  an outlier with respect to  $T$ . The local class outlier detection problem is to find those suspicious objects according to the value of  $LCOF(p)$ .

**Definition 3. (Reference Class Outlier Detection).** Given  $DB$  and a target set  $T$ , for an object  $p \in T$ , the reference class outlier factor of  $p$ , denoted as  $RCOF(p, C_i)$ , captures the degree to which we call  $p$  an outlier with respect to  $C_i$ , the set of objects with class label  $cl_i$ . The reference class outlier detection problem is to find those suspicious objects according to the value of  $RCOF(p, C_i)$ .

In Definition 2, we consider the problem of detecting outlying observations from a target set of objects  $T$ , with respect to itself. It is named as “local class outlier” because even though everything may look “normal” in  $DB$  when we ignore object types, there is still a possibility that the objects are “suspicious” objects if we relate them only to objects of  $T$ .

In Definition 3, we want to discover points in  $T$  that “arouse suspicions” with respect to points in set  $C_i$ . Note that our notion of reference class outlier is same as that

of cross-outlier proposed in [2]. As pointed out in [2], the local class outliers are a special case of reference class outlier, when  $C_i=T$ . However, the following observations motivate us to define them separately:

1) Firstly, from the application viewpoint, in spite of their similarities in representation, their meanings differ significantly. Consider Example 1, a local class outlier in the loyal segment is a possible churn customer that needs to be retained. While a reference class outlier in the vulnerable segment may be a loyal customer because it shares little in common with customers in the vulnerable segment. Hence, explicitly distinguishing these two kinds of outliers is of great advantage for business users.

2) Secondly, from the viewpoint of the algorithm design, some traditional approaches (or local class outlier detection methods) can be modified to deal with the reference class outlier detection problem, but the task is non-trivial [2].

So far, we only provide the concepts of local class outlier and reference class outlier, and assume that both can be mined by extending the existing methods. Implantation details will be discussed in the next section. We further assume that the outlier factor values are normalized onto the interval  $[0,1]$ . For the *degree* of an object being an outlier measured by its outlier factor, without loss of generality, we assume that the higher the outlier factor of an object is, the higher its degree of being an outlier. In the sequel, we introduce **set union class outlier**, where the reference set is the union of some sets.

**Definition 4. (Set Union Class Outlier).** Given  $DB$  and a target set  $T$ , for an object  $p \in T$ , the set union class outlier factor of  $p$  with respect to  $R$ , is denoted as  $SUCOF(p, R)$ , where  $R \in Sp(DB)$ . Moreover, a set union class outlier can be divided into two kinds: *local set union class outlier* and *reference set union class outlier*. That is,  $p$  is called a local set union class outlier if  $T \subseteq R$ ; its outlier factor is denoted as  $LSUCOF(p, R)$ ; Otherwise,  $p$  is called a reference set union class outlier and its outlier factor is denoted as  $RSUCOF(p, R)$ .

According to Definition 4, both local class outlier and reference class outlier are special cases of set union class outlier. They are distinguished by the reference set. More concisely, if  $R=T$ ,  $LSUCOF(p, R)=LCOF(p)$ ; if  $C_i=R$ ,  $RSUCOF(p, R)=RCOF(p, R)$ ; More importantly, when  $R=DB$ , the problem of set union class outlier detection is equal to the traditional single-class outlier detection problem. That is, given a set of objects, find those that deviate significantly from the rest regardless of class labels. Therefore, all aforementioned outlier detection problems can be studied in the framework of set union class outlier.

We now turn to another kind of outlier-**combined class outlier**, which can be regarded as a further extension to the set union class outlier.

**Definition 5. (Combined Class Outlier).** Given  $DB$  and a target set  $T$ , for an object  $p \in T$ , the combined class outlier factor of  $p$  is defined as:

$$CCOF(p, \{R_1, R_2, \dots, R_m\}) = \oplus (v_1, v_2, \dots, v_m)$$

Where  $\oplus$  is the *combiner* function, which computes the outlier factor value form  $v_1, v_2, \dots, v_m$ ,  $R_i \in Sp(DB)$  and  $v_i = SUCOF(p, R_i)$  or  $v_i = 1 - SUCOF(p, R_i)$ .

If  $m > 1$ , we call  $p$  a *true combined class outlier*.

From Definition 5, the value of the combined class outlier factor is the result of applying the *combiner* function  $\oplus$  to some set of values. Each  $v_i \in (v_1, v_2, \dots, v_m)$  is the set union class outlier factor of  $p$  or its reciprocal.

Compared to other types of class outlier, we refer to multiple sets instead of a single set. Note that the set union class outlier is also a special case of the combined class outlier, that is,  $CCOF(p, \{R_i\}) = \bigoplus (v_i) = v_i = SUCOF(p, R_i)$  or  $1 - SUCOF(p, R_i)$ .

To complete the description of the combined class outlier, we discuss the following issues: our choice of combining operators and the usefulness of combined class outlier in real applications.

**Choosing a combining operator:** Our potential choices for  $\bigoplus$  are the following<sup>1</sup>. We offer some additional insights on these choices in Section 5.

- The product operator  $\Pi : \bigoplus (v_1, v_2, \dots, v_m) = v_1 v_2 \dots v_m$ .
- The addition operator  $+$ :  $\bigoplus (v_1, v_2, \dots, v_m) = v_1 + v_2 + \dots + v_m$ .
- A generalization of addition operator—it is called the  $S_q$  combining rule, where  $q$  is an odd natural number.  $S_q(v_1, v_2, \dots, v_m) = (v_1^q + v_2^q + \dots + v_m^q)^{(1/q)}$ . Note that the addition is simply an  $S_1$  rule.
- A “limited” version of  $S_q$  rules, denoted as  $S_\infty$ .  $S_\infty(v_1, v_2, \dots, v_m)$  is defined to be equal to  $v_i$ , where  $v_i$  has the largest absolute value among  $(v_1, v_2, \dots, v_m)$ .

Thus, the  $S_1$  combining rule is *linear* to the component outlier factors. On the other hand, for  $q > 1$ , the  $\Pi$  and  $S_q$  rules involve a *non-linear* term for each individual outlier factor.  $S_\infty$  is an especially appealing rule, since it is non-linear and particularly fast to compute. It has certain useful “sum-like” properties.

**Practical use of the combined class outlier:** According to Hawkins [3], an outlier is an observation that deviates so much from other observations as to arouse suspicion that it was generated by a different mechanism. We introduce a set of class outlier factors for each object in the dataset, indicating its degree of being an outlier. We have assumed that the outlier factor values are normalized onto the interval  $[0, 1]$  and the higher the outlier factor value of an object  $p$  is, the higher the degree of being an outlier (For set union class outlier, its outlier factor is denoted as  $SUCOF(p, R_i)$ ). Therefore, we give the following statement:

*“The lower the outlier factor value of an object  $p$  is, the lower the degree of being an outlier. More concisely, objects with lower values of outlier factor are more likely to be generated by the same mechanism”.*

Based on the above statement and assumption, a higher  $SUCOF(p, R_i)$  value (or a lower  $(1 - SUCOF(p, R_i))$  value) indicates a higher chance of being an outlier. In contrast, a lower  $SUCOF(p, R_i)$  value (or a higher  $(1 - SUCOF(p, R_i))$  value) indicates a lower chance of being an outlier.

Furthermore, given two sets  $R_1$  and  $R_2$ ,  $SUCOF(p, R_1)$  and  $SUCOF(p, R_2)$  denote the outlier factor values of an object  $p$  with respect to  $R_1$  and  $R_2$ , respectively. From the outlier detection viewpoint, all possible combinations using the combining operator  $\bigoplus$  are listed in Table 2.

As shown in Table 2, taking  $SUCOF(p, R_1) \bigoplus SUCOF(p, R_2)$  as an example, we can derive the fact that: “The higher the  $SUCOF(p, R_1) \bigoplus SUCOF(p, R_2)$  value of the

<sup>1</sup> The choices for  $\bigoplus$  have been widely studied in dynamic systems. In this paper, we borrow some interesting idea from Ref. [4].

object  $p$  is, the higher the chance of being an outlier with respect to  $R_1$  and  $R_2$ ". Other combinations can be illustrated in a similar way. To complete the description of the usefulness of the combined class outlier, considering the following example.

**Table 2.** Combinations and their meanings

Combinations	Meanings
$SUCOF(p, R_1) \oplus SUCOF(p, R_2)$	The object $p$ deviates from both $R_1$ and $R_2$ .
$SUCOF(p, R_1) \oplus (1-SUCOF(p, R_2))$	The object $p$ deviates from both $R_1$ and looks normal in $R_2$ .
$(1-SUCOF(p, R_1)) \oplus SUCOF(p, R_2)$	The object $p$ deviates from both $R_1$ and looks normal in $R_2$ .
$(1-SUCOF(p, R_1)) \oplus (1-SUCOF(p, R_2))$	The object $p$ looks normal in both $R_1$ and $R_2$ .

**Example 2.** Following Example 1, a potential customer that was transferred from the loyal segment (denoted as  $L$ ) to the vulnerable segment (denoted as  $V$ ) would satisfy the following desired properties. 1) His (Her) behavior is very exceptional in comparison with the rest of customers in the loyal segment, and 2) he (she) looks very normal with respect to customers in the vulnerable segment. Hence, for our purpose, to simultaneously incorporate these two measures into an integrated measure,  $SUCOF(p, L) \oplus (1-SUCOF(p, V))$  will be a good choice.

The "semantic outlier" proposed in [1] is defined informally as: "A semantic outlier is a data point, which behaves differently with other data points in the same class, but looks "normal" in another class". Obviously, the "semantic outlier" is a special case of combined class outlier.

### 3 Algorithms

In this section, we propose our methods for mining class outliers. We begin with identifying the difference on the score model building between the class outlier detection and the traditional outlier detection. Consequently, we show how to modify existing outlier detection algorithms for class outlier mining.

#### 3.1 Difference on Score Model Building

In Section 2, we have discussed that the outlier factor of the combined class outlier is result of applying the *combiner* function  $\oplus$  to some set of values of set union class outlier factors. Therefore, the main challenge task is to compute the set union class outlier factor for each object in the target set. That is, we focus on the problem of detecting outlying observations from a target set of points  $T$ , with respect to a reference set of points  $R$ . We want to discover points  $p \in T$  that "arouse suspicions" with respect to points  $r \in R$ . As discussed in Section 2, the traditional single-set outliers are a special case of  $R = T$ .

Since the traditional outlier detection algorithms build a scoring model using data points in  $T$  and the outlier factor for each point is computed with the derived model, we can describe those algorithms using the following schema  $A1$  (See Fig.1).

The algorithmic schema  $A2$  for the class outlier detection algorithms is shown in Fig.2. Compared with schema  $A1$ , the training set for building the scoring model is  $R + \{p\}$ . Thus, for each point  $p$ , the model  $SM$  has to be re-constructed. For real data

mining applications, the size of dataset is often very large, the frequency re-constructing process will be time-consuming. Motivated by the above observation, as shown in Fig.3, we propose a new algorithmic schema A3 by modifying the algorithmic schema A2.

The algorithmic schema A3 can be regarded as a fast approximation schema for the class outlier detection. In Section 3.2 and Section 3.3, we modify two existing outlier detection algorithms according to schema A3.

The algorithmic schema A3 can be regarded as a fast approximation schema for the class outlier detection. In Section 3.2, we modify two existing outlier detection algorithms according to schema A3.

**Algorithmic Schema A1:**  
Build scoring model  $SM$  using  $T$   
foreach Point  $p \in T$  {  
Outlier factor  $OF(p) = \text{ComputeScore}(p, SM)$ ;  
}  
Sort points according to outlier factors  
DoSomething ();

**Fig. 1.** The algorithmic schema A1

**Algorithmic Schema A2:**  
foreach Point  $p \in T$  {  
Build scoring model  $SM$  using  $R + \{p\}$ ;  
Outlier factor  $OF(p) = \text{ComputeScore}(p, SM)$ ;  
}  
Sort points according to outlier factors  
DoSomething ();

**Fig. 2.** The algorithmic schema A2

**Algorithmic Schema A3:**  
Build scoring model  $SM$  using  $R$   
foreach Point  $p \in T$  {  
Outlier factor  $OF(p) = \text{ComputeScore}(p, SM)$ ;  
}  
Sort points according to outlier factors  
DoSomething ();

**Fig. 3.** The modified algorithmic schema A3 for the class outlier detection algorithms

### 3.2 Two Class Outlier Detection Algorithms

According to the algorithmic schema A3, We make modifications to two traditional outlier detection algorithms, i.e., the frequent pattern based outlier detection method [5] and the cluster-based local outlier detection approach [6]. Due to space limitation, a detailed description on the two modified algorithms, *FindCCBLOF* and *FindCFPOF*, is omitted here and can be found in [7].

## 4 Applications in CRM

In this section, we describe two potential applications of our method in CRM: Loyalty and Direct marketing.

### 4.1 Loyalty

Loyalty is a key program of CRM. It is important to detect customers, which have the intension to leave from the loyal segment (denoted as  $L$ ) to the vulnerable segment (denoted as  $V$ ). As discussed in Section 2, these customers satisfy the following properties.

1. Their behaviors are very exceptional when compared with the rest of customers in the loyal segment.
2. These customers look similar to the customers in the vulnerable segment.

Hence, within the framework of class outlier, we will have three alternative measures to identify those customers (a customer is denoted as  $p$ ).

1.  $SUCOF(p, L)$  –measure the likelihood of  $p$  being an outlier with respect to  $L$ .
2.  $(1-SUCOF(p, V))$  –measure the likeness of  $p$  with respect to  $V$ .
3.  $SUCOF(p, L) \oplus (1-SUCOF(p, V))$  –measure both (1) and (2)

In summary, we apply class outlier factors as loyalty scores to detect customers that have the likelihood to leave the loyalty segment. For these identified customers (outliers), the marketer can design suitable retention program to keep them.

## 4.2 Direct Marketing

In direct marketing, models (profiles) are generated to select potential customers to promote a given product from similar campaigns.

Let  $U$  be a finite set of objects. Elements of  $U$  may be customers or products we are interested in market oriented decision making[8]. The set  $U$  is divided into three pair-wise disjoint classes, i.e.,  $U = P \cup N \cup T$ . The sets  $P$ ,  $N$  and  $T$  are called *positive*, *negative* and *targeting* (or *don't know*) instances, respectively.  $P$  is the set of current customers, which is often the minority class, e.g., the responding class in direct marketing.  $N$  is the set of people who did not responded to the previous marketing campaign.  $T$  is the target set of customers who have a potential to respond to the campaign to be launched. A direct marketing problem may be defined as finding elements from  $T$ , and possibly from  $N$ , that are similar to elements in  $P$ , and possibly dissimilar to elements in  $N$ . In other words, the goal is to identify elements from  $T$  and  $N$  that are more likely to become new members of  $P$ . We are interested in developing a scoring model with class outliers to rank elements of  $T$  accordingly.

The analysis of a direct marketing campaign entails two stages. First, *feature selection* determines the variables that are relevant to the specific target selection problem. Second, the rules for selecting the customers should be determined, given the relevant features [9]. At present, statistical tools like CHAID [10] are often used to search for features that discriminate between respondents and non-respondents, and to build decision tree models for target selection.

The problem of direct marketing has received a great deal of attention [e.g., 8, 9, 11,12]. In this paper, the class outlier factor is utilized as a scoring measure to score the data points in  $T$ . Since potential responders in  $T$  are similar to elements in  $P$ , possibly dissimilar to elements in  $N$ , and possibly dissimilar to elements in its own set  $T$ . Thus, from the viewpoint of class outlier detection, there can be many alternative ways to design methods to score the data. For example, we can simply score each data case  $p$  using  $1-SUCOF(p, P)$ . This method is reasonable because the set union class outlier factor value can be seen as a probability estimate indicating the likelihood that the case belongs to the positive class.

However, it is not straightforward to designing the best scoring method based on class outlier because of different alternatives. In Table 3, we list all the possible scoring functions and their names.  $W_T$ ,  $W_P$  and  $W_N$  are the weights of target class, positive class and negative class, respectively.



The first difficulty is the selection of a function to be used. To answer this question, the following factors should be considered:

- The availability of datasets. For example, in some cases, the sets  $P$  or  $N$  are not available so that it appears that the  $TSF$  function will be the only choice for user.
- The data quality of datasets

Real life datasets contain missing values and incorrect data resulted from human, measurement and computation errors. Such data will affect the results of data mining algorithms significantly. Therefore, it will be better to select scoring functions that involve confident datasets only.

In general, the scoring function  $PTSF$  always produces reliable result, which is verified in the experimental evaluation section.

**Table 3.** Class Outlier Based Scoring Functions for Direct marketing

Scoring Functions Using Class Outlier	Names
$1-SUCOF(p, P)$	PSF
$SUCOF(p, N)$	NSF
$SUCOF(p, T)$	TSF
$W_N * SUCOF(p, N) \oplus W_P * (1-SUCOF(p, P))$	PNSF
$W_P * SUCOF(p, N) \oplus W_T * SUCOF(p, T)$	NTSF
$W_T * SUCOF(p, T) \oplus W_P * (1-SUCOF(p, P))$	PTSF
$W_T * SUCOF(p, T) \oplus W_P * (1-SUCOF(p, P)) \oplus W_N * SUCOF(p, N)$	PNTSF

Now the problem is what should be the weights in those functions based on the truly combined class outliers. Since we have to consider the two factors that affect the choice of functions discussed above, the weights must reflect their needs. In general, the weight term of the less confident dataset should be small because we would like to reduce its effect on the scores. In contrast, the weight term of the more confident dataset should be large.

### 5 Empirical Evaluations

We applied the proposed class outlier detection method to the “Coil Challenge 2000” dataset [13]. The training set contains over 5000 descriptions of customers, including the information of whether or not they have a caravan insurance policy.

The prediction task in Coil Challenge 2000 [14] is to find the subset of customers with a probability of having a caravan insurance policy above some boundary probability. The task was formulated more precisely as the demand for 20 percent of the 4000 test data record indexes. These 800 customers should be likely to be interested in a caravan insurance policy such that a virtual mail campaign for these addressees would be as successful as possible.

The Coil data set encloses 85 highly correlated input variables. Feature selection is the first important task. In this paper, we employ the feature selection method proposed in [15]. The variables {59, 47, 25, 82, 65} are suggested as relevant variables for the problem in [15]. Therefore, only these five attributes are utilized in our mining process. As discussed in Section 4.2, the training set is divided into  $P$  and  $N$ , and the test dataset is taken as  $T$ . In our experiments, we let all class outlier based scoring

methods to produce top 800 outliers (customers), and examine the number of policy owners in the chosen set of 800 customers. The class outlier detection algorithms *FindCCBLOF* and *FindCFPOF* are tested to examine their effectiveness. For all the experiments, parameters needed by the *FindCCBLOF* algorithm are set to default as suggested in [5]. For the *FindCFPOF* algorithm, the parameter *mini-support* for mining frequent patterns is fixed to 2%, and the maximal number of items in an itemset is set to 5. Table 4 shows the experimental results and some important observations are summarized as follows:

(1) The scores achieved by the 43 individuals or teams that submitted entries to the contest are summarized as follows [16]. The winning entry identified 121 caravan policyholders among its 800 top predictions. The next best methods identified 115 and 112 policyholders. The mean score was 95.4, with a standard deviation of 19.4. The most common score was 109, and the median score was 103 [16].

According to Table 4, our first claim is that the performance of class outlier based method is comparable to other popular techniques in the direct marketing applications. It should be noted that the results of scoring functions *NSF*, *PSF* and *TSF*, which only use one dataset, are also included. Obviously, it can be known in advance that these functions will not produce satisfactory results, because they utilize only one dataset (Furthermore, the NSF function explores the test dataset only). Thus, if we don't consider these functions, a better average performance can be achieved.

**Table 4.** Experimental Results on Coil Challenge 2000 Dataset

Scoring Function		Number of True policy Owners	
		FindCFPOF	FindCCBLOF
[TSF] <i>SUCOF</i> ( $p, T$ )		62	87
[PSF] 1- <i>SUCOF</i> ( $p, P$ )		80	86
[NSF] <i>SUCOF</i> ( $p, N$ )		76	76
[PTSF] $W_T * \text{SUCOF}(p, T) \oplus$ $W_P * (1 - \text{SUCOF}(p, P))$	$\oplus = "+"$	94	94
	$\oplus = "\prod"$	110	99
	$\oplus = "S_q", q=3$	101	108
	$\oplus = "S_\infty"$	86	99
[NTSF] $W_T * \text{SUCOF}(p, T) \oplus$ $W_N * \text{SUCOF}(p, N)$	$\oplus = "+"$	65	92
	$\oplus = "\prod"$	65	94
	$\oplus = "S_q", q=3$	65	91
	$\oplus = "S_\infty"$	76	94
[PNSF] $W_N * \text{SUCOF}(p, N) \oplus$ $W_P * (1 - \text{SUCOF}(p, P))$	$\oplus = "+"$	111	97
	$\oplus = "\prod"$	111	99
	$\oplus = "S_q", q=3$	102	97
	$\oplus = "S_\infty"$	85	94
[PNTSF] $W_T * \text{SUCOF}(p, T) \oplus$ $W_P * (1 - \text{SUCOF}(p, P)) \oplus$ $W_N * \text{SUCOF}(p, N)$	$\oplus = "+"$	104	102
	$\oplus = "\prod"$	106	95
	$\oplus = "S_q", q=3$	89	101
	$\oplus = "S_\infty"$	84	100

(2) How does the choice of scoring functions affect the results? From the results reported in Table 7, it is very clear that the scoring functions contain the combine operator ( $\oplus$ ) outperformed those simple functions. This is because *true combined class outlier* based scoring functions utilize more available datasets. However, it

needs to point out that, simple scoring functions are also useful and interesting in their own rights in practice. For example, when the positive dataset ( $P$ ) and negative dataset ( $N$ ) are not available to the analyzer, that is, the direct marketing campaign is in its initial stage, the  $TSF$  function will be the only applicable function. It is also one important advantage of our class outlier based method that we can still detect potential customers when only the test dataset is available.

For those complicated scoring functions with combine operator ( $\oplus$ ) as their component, they exhibit comparable performances. While in average, the  $PNSF$  function is a little better than the other functions. Hence, the  $PNSF$  function is more preferable than other alternatives.

(3) How does the choice of combine operator ( $\oplus$ ) affect the results? From Table 7, the “+” operator and “ $\Pi$ ” operator are the clear winners. That is, for a specified scoring function, the “+” operator and “ $\Pi$ ” operator outperformed  $S_q$  and  $S_\infty$  in most cases. This observation suggests that the “+” operator and “ $\Pi$ ” operator will be a better choices in practice for users. Moreover, further experiments show that with increase of  $q$  in the  $S_q$  operator, the performance of scoring functions will deteriorate.

In summary, the experimental results show that the class outlier method is competitive in the direct marketing applications. Furthermore, it is more flexible than other popular techniques because different kinds of scoring functions are available for various situations, even for the situation that other method is failed to work (It is the case that *only* test dataset is available).

## 6 Conclusions

In this paper, we formulated the problem of the class outlier detection and presented effective algorithms for outlier mining. Furthermore, its potential applications in CRM are discussed. The effectiveness of our method is also empirically studied.

## Acknowledgements

The High Technology Research and Development Program of China (No. 2002AA413310, No. 2003AA4Z2170, No. 2003AA413021), National Nature Science Foundation of China (No. 40301038) and IBM SUR Research Fund supported this research.

## References

- [1] Z. He, S. Deng, X. Xu. Outlier detection integrating semantic knowledge. In: WAIM'02, pp.126-131, 2002.
- [2] S. Papadimitriou, C. Faloutsos. Cross-outlier detection. In: Proc of SSTD'03, pp.199-213, 2003.
- [3] D. Hawkins. Identification of outliers. Chapman and Hall, Reading, London, 1980.

- [4] D. Gibson, et al. Clustering categorical data: an approach based on dynamic systems. VLDB'98, pp, 1998.
- [5] Z. He, et al.. A Frequent Pattern Discovery Method for Outlier Detection. WAIM'04, 2004
- [6] Z. He, X. Xu, S. Deng. Discovering Cluster Based Local Outliers. Pattern Recognition Letters, 2003.
- [7] Z. He, J. Huang, X. Xu, S. Deng. Mining Class Outlier: Concepts, Algorithms and Applications. Technology Report, HIT, 2003.  
[http://www.angelfire.com/mac/zengyouhe/publications/Class\\_Outlier.pdf](http://www.angelfire.com/mac/zengyouhe/publications/Class_Outlier.pdf).
- [8] Y. Yao, N. Zhong, J. Huang, C. Ou, C. Liu. Using Market Value Functions for Targeted Marketing Data Mining. International Journal of Pattern Recognition and Artificial Intelligence, 2002, 16(8): 1117-1132.
- [9] M. Setnes, U. Kaymak. Fuzzy Modeling of Client Preference from Large Data Sets: An Application to Target Selection in Direct Marketing. IEEE Transactions on Fuzzy Systems, 2001, 9(1): 153-163.
- [10] SPSS Inc., *SPSS CHAID for Windows 6.0*. Englewood Cliffs, NJ: Prentice-Hall, 1993.
- [11] C.X. Ling , C. Li. Data Mining for Direct Marketing: Problems and Solutions. KDD'98, pp.73- 79, 1998.
- [12] B. Liu, Y. Ma, C. K. Wong, P. S. Yu. Scoring the Data Using Association Rules. Applied intelligence, 2003.
- [13] The Coil dataset can found at: <http://www.liacs.nl/~putten/library/cc2000/>.
- [14] <http://www.dcs.napier.ac.uk/coil/challenge/thetasks.html>.
- [15] A. Lewandowski. How to detect potential customers. In: CoIL Challenge 2000: The Insurance Company Case, Technical Report 2000-09, Leiden Institute of Advanced Computer Science, Netherlands, 2000.
- [16] C. Elkan. Magical Thinking in Data Mining: Lessons From CoIL Challenge 2000. In: Proc of KDD'01, 2001.

# CD-Trees: An Efficient Index Structure for Outlier Detection\*

Huanliang Sun<sup>1,2</sup>, Yubin Bao<sup>1</sup>, Faxin Zhao<sup>1</sup>, Ge Yu<sup>1</sup>, and Daling Wang<sup>1</sup>

<sup>1</sup> School of Information Science and Engineering, Northeastern University,  
Shenyang 110006, P.R.China  
{baoyb, yuge}@mail.neu.edu.cn

<sup>2</sup> Shenyang Architectural and Civil Engineering University, Shenyang 110015, P.R.China

**Abstract.** Outlier detection is to find objects that do not comply with the general behavior of the data. Partition is a kind of method of dividing data space into a set of non-overlapping rectangular cells. There exists very large data skew in real-life datasets so that partition will produce many empty cells. The cell-based algorithms for outlier detection don't get enough attention to the existence of many empty cells, which affects the efficiency of algorithms. In this paper, we propose the concept of Skew of Data (SOD) to measure the degree of data skew, and which approximates the percentage of empty cells under a partition of a dataset. An efficient index structure called CD-Tree and the related algorithms are designed. This paper applies the CD-Tree to detect outliers. Compared with cell-based algorithms on real-life datasets, the speed of CD-Tree-based algorithm increases 4 times at least and that the number of dimensions processed also increases obviously.

## 1 Introduction

An outlier is an observation that deviates so much from other observations as to arouse suspicion that it was generated by a different mechanism [1]. Outlier detection aims to discover outliers. Many applications have these needs, such as intrusion detection, telecom and credit card fraud detection, loan approval, weather prediction, etc.

Konrr and Ng[2] proposed an efficient approach for outlier detection. They defined outlier as follows: "An object  $O$  in a dataset  $X$  is a  $DB(\delta, d)$ -outlier if at least percentage  $\delta$  of the objects in  $X$  lies greater than distance  $d$  from  $O$ ." The main benefit of the approach is that it does not require any apriori knowledge of data distributions. The cell-based algorithm is the most important contribution in the paper, which is based on partitioning the space into a uniform grid of cells and then using these cells to detect outliers. The algorithm doesn't work effectively for high dimensions or fine granularity partition because the number of cells is too many. In fact, the algorithm doesn't pay enough attention to the existence of empty cells, which affects the efficiency of algorithms. So, we present the concept of Skew of Data (SOD). The SOD

---

\* Supported by the National Natural Science Foundation of China under Grant No.60173051, the Teaching and Research Award Program for Outstanding Young Teachers in Higher Education Institutions of the Ministry of Education, China.

approximates the percentage of empty cells under a partition, and the value of SOD in most of real-life datasets is very large. We present an efficient index structure called CD-Tree, which only stores the non-empty cells, and apply it to detect outliers. The contributions of this paper are as follows:

- We propose the concept of Skew of Data (SOD), and give the method to calculate SOD, and discuss the relationship between SOD and data partition.
- We present an efficient index structure called CD-Tree(Cell Dimension Tree) and some efficient algorithms.
- We apply CD-Tree to outlier detection. The experimental results show that the speed of CD-Tree-based algorithm increases 4 times at least than the cell-based algorithms and that the maximum number of dimensions that can be processed also increases obviously.

The rest of this paper is organized as follows. Section 2 discusses related work. Section 3 presents the theory of Skew of Data. Section 4 presents CD-Tree. Section 5 discusses the outlier detection algorithms based on CD-Tree. In section 6 we present the performance of the algorithm. Finally, section 7 concludes the paper.

## 2 Related Work

Until now the algorithms of outlier detection can be classified as follows: statistics-based algorithms[3], distance-based algorithms[2, 4, 5], density-based algorithms[6], and high dimension algorithms [7], and so on.

In statistics, the data objects are modeled using a statistical distribution (such as Gaussian Distribution), and outliers are the points that deviate away from the model. The defect of this method is that the distribution of data must obey some standard distribution, but the real-life data often don't fit any standard mathematic distribution. In order to solve this problem, Knorr and Ng[2] propose the distance-based algorithm. Except for cell-based algorithm, the authors also propose two simple algorithms, one is the Nested-Loop algorithm (NL) with complexity  $O(k*N^2)$ , where  $k$  is the dimension of data, and  $N$  is the size of dataset. Another one is index-based algorithm with complexity  $O(k*N^2)$ .

Ramaswamy et al[5] give another distance-based outlier detection algorithm. They also provide a partition-based algorithm. It uses a clustering algorithm to partition dataset, then detect outlier in partitions. This algorithm can present the level of outliers, but it depends on clustering algorithms, so the efficiency of this algorithm is influenced. Aggarwal and Yu[7] discusses a method to deal with the high-dimension outliers. Breunig et al[6] present local-outlier detection algorithm.

## 3 Skew of Data (SOD)

The result of data partition is a set of cells. If the granularity of partition is fine or the number of dimensions is high, partition will produce too many cells. In fact there

exist many empty cells that are useless for data analysis. We introduce the SOD concept in order to analyze the relationship between partition and data skew. SOD is used for measuring the degree of data skew. Through calculating SOD, we can estimate the proportion of empty cells under a partition.

### 3.1 The Concept of SOD

Let  $A = \{A_1, A_2, \dots, A_k\}$  be a set of bounded, totally ordered domains and  $S = A_1 \times A_2 \times \dots \times A_k$  be a  $k$ -dimensional numerical space. We refer to  $A_1, A_2, \dots, A_k$  as the dimensions(attributes) of  $S$ . The dataset consists of a set of  $k$ -dimensional points— $X = \{x_1, x_2, \dots, x_N\}$  where  $x_i = \langle x_{i1}, x_{i2}, \dots, x_{ik} \rangle$ . The  $j$ th component of  $x_i$ ,  $x_{ij}$  is drawn from domain  $A_j$ .

Let  $P$  be a set of non-overlapping rectangular cells, which is obtained by partitioning every dimension into equal length. Each cell is the intersection of one interval from each attribute. It has the form  $\{c_1, c_2, \dots, c_k\}$  where  $c_i = [l_i, h_i)$  is a right-open interval in the partitioning of  $A_j$ . A cell can also be denoted as  $(cNO_1, cNO_2, \dots, cNO_k)$ , where  $cNO_i$  is the interval number of the cell on  $i$ -th dimension from 1 to the total number of interval. We call  $P$  a partition of dataset  $X$ .

**Definition 1. (SOD( $X, P$ ) of dataset  $X$  under partition  $P$ )** The  $SOD(X, P)$  of dataset  $X$  under partition  $P$  is defined as  $\frac{1}{2N} \left( \sum_{i=1}^m |p_i - t| \right)$  where  $p_i$  is the number of points in cell  $C_i$ ,  $m$  is the number of cells and  $N$  is the total number of data points. In addition,  $t$  is a constant which is the average of points number in cells calculated by formula  $t = N/m$ .

The  $SOD(X, P)$  of a dataset is a value in  $[0, 1]$ . If  $SOD(X, P) = 0$ , then there is no skew under partition  $P$ . The closer to 1  $SOD(X, P)$  is, the larger the data skew is.

**Theorem 1:** The SOD of a dataset under partition  $P$  is equal to  $\frac{1}{N} \left( \sum_{j=1}^v (p_j - t) \right)$  where  $p_j$  is the number of points in  $C_j$  in which the number of points is larger than  $t$ .

**Proof:** By definition 1,  $SOD(X, P) = \frac{1}{2N} \left( \sum_{i=1}^m |p_i - t| \right)$ , suppose the number of cells in which point's number are larger than  $t$  is equal to  $v$ , the total number of all cells is  $m$ . Then,

$$\begin{aligned} SOD(X, P) &= \frac{1}{2N} \left( \sum_{j=1}^v (p_j - t) - \sum_{i=1}^{m-v} (p_i - t) \right) = \frac{1}{2N} \left( \sum_{j=1}^v p_j - vt - \sum_{i=1}^{m-v} p_i + (m-v)t \right) \\ &= \frac{1}{2N} \left( 2 \sum_{j=1}^v p_j - \left( \sum_{j=1}^v p_j + \sum_{i=1}^{m-v} p_i \right) - 2vt + mt \right). \quad \text{Because } \sum_{j=1}^v p_j + \sum_{i=1}^{m-v} p_i = \sum_{i=1}^m p_i = N \quad \text{and} \\ &mt = N. \text{ thus, } \frac{1}{2N} \left( 2 \sum_{j=1}^v p_j - N - 2vt + N \right) = \frac{1}{N} \left( \sum_{j=1}^v (p_j - t) \right). \end{aligned}$$

By theorem 1, when calculating SOD of a dataset, we only need to visit the cells in which the number of points is larger than  $t$ . So it makes calculation of SOD simple.

**Definition 2. (Sub-partition)** Let  $P_1$  and  $P_2$  be two partitions of dataset  $X$ , if  $\forall c_1 \in P_1$ , then  $\exists c_2 \in P_2$ , such that  $c_1 \subset c_2$ , i.e. for the interval  $[l_{1i}, h_{1i})$  of the cell  $c_1$ , then  $\exists [l_{2i}, h_{2i})$  of  $c_2$ , such that  $[l_{1i}, h_{1i}) \subset [l_{2i}, h_{2i}), i = 1, \dots, k$ . We call  $P_1$  a sub-partition of  $P_2$ , denoted as  $P_1 \subset P_2$ .

**Theorem 2:** Given two partitions  $P_1$  and  $P_2$  of dataset  $X$ , if  $P_1 \subset P_2$ , then  $SOD(X, P_1) \geq SOD(X, P_2)$ .

**Proof:** Let  $m_1, m_2$  be the total number of cells of partition  $P_1$  and  $P_2$  respectively, and  $t_1, t_2$  be the average points number of them. Because of  $P_1 \subset P_2$ , then  $m_1 = r m_2$ ,  $t_2 = r t_1$  where  $r$  is a natural number. By definition 1:

$$SOD(X, P_2) = \frac{1}{2N} \left( \sum_{i=1}^{m_2} |p_i - t_2| \right), \text{ any cell } c_i \in P_2 \text{ can be divided into } r \text{ cells in } P_1.$$

$$\text{i.e. } |p_i - t_2| = \left| \sum_{j=1}^r p_{ij} - r t_1 \right| \leq \sum_{j=1}^r |p_{ij} - t_1|. \text{ Then}$$

$$SOD(X, P_2) \leq \frac{1}{2N} \left( \sum_{i=1}^{m_2} \sum_{j=1}^r |p_{ij} - t_1| \right) = \frac{1}{2N} \left( \sum_{i=1}^{r m_2} |p_i - t_1| \right), \text{ i.e. } SOD(X, P_1) \geq SOD(X, P_2).$$

Theorem 2 shows that the finer the granularity of a partition is, the larger SOD is. We can use the SOD under coarse granularity to estimate the SOD under fine granularity because a big interval partition can be performed effectively. In addition, a SOD can express the degree of fullness of the space filled with data, so the SOD is approximately equal to the percentage of empty cells. Suppose two partitions  $P_1, P_2$  of dataset  $X$ ,  $P_1 \subset P_2$ , and let  $n_1, n_2$  denote the non-empty cell number of  $P_1$  and  $P_2$  respectively. Because the percentage of empty cells approximates SOD, then  $n_1 \approx (1 - SOD(X, P_1)) \times m_1$  and  $n_2 \approx (1 - SOD(X, P_2)) \times m_2$  where  $m_1$  and  $m_2$  are the total number of cells under two partitions. We can partition dataset under  $P_2$  firstly, and then by theorem 2, there exists  $SOD(X, P_1) \geq SOD(X, P_2)$ , thus we can estimate  $n_1$  by  $n_1 \leq (1 - SOD(X, P_2)) \times m_1$ . After the upper bound of  $n_1$  is estimated, we can estimate the space cost to choose a suitable partition.

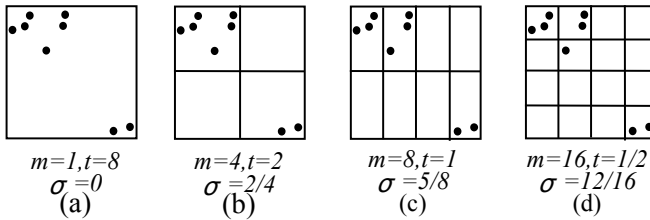


Fig. 1. The SOD in different partitions



In Fig. 1, there is an example to illustrate the concept of SOD, theorem 1 and 2. There are four partitions that divide data space into 1, 4, 8, and 16 cells. Let  $m$ ,  $t$  and  $\sigma$  denote the number of cells, the average selectivity of cells and SOD's value of a dataset under a partition. In Fig. 1(a), there is only one cell, so  $\sigma$  is equal to 0 by SOD definition. In Fig. 1(b)(c)(d),  $\sigma$  turns larger as the partition turns finer. The value  $\sigma$  is  $1/2$ ,  $5/8$ ,  $6/8$  respectively, and equal to the percentage of empty cells respectively.

3.2 SOD of Typical Datasets

In this section, we calculate the SOD of some UCI datasets [8] and the other typical datasets (NHL datasets in [2], NBA datasets in [5]). We divide each dimension into four intervals. Let  $\sigma$ ,  $q$  denote the SOD of a dataset and the percentage of empty cells under a partition. As Table 1 shows, the SOD of most of these real-life datasets are large, and the higher dimension number is, the larger the SOD is.

Table 1. The SOD of typical datasets

Dataset name	Dimen -sionality	Size	The total num- ber of cells	Non-empty cells	$\sigma$	$q$
NHL96	3	855	$4^3$	14	0.881	0.781
NBA97-98	5	329	$4^5$	42	0.958	0.958
Auto-mpg	6	398	$4^6$	65	0.984	0.984
Liver-disorder	7	345	$4^7$	89	0.994	0.994
Cpu-performance	8	209	$4^8$	22	0.999	0.999
NBA97-98	14	184	$4^{14}$	176	0.999	0.999
Image-segmentation	19	210	$4^{19}$	25	0.999	0.999

4 CD-Tree

In most of real-life datasets there exists large SOD, so it isn't necessary to store empty cells. We seek an index structure to represent cell structure. Such a data structure should (a) retain the essential information of all non-empty cells; (b) enable efficiently to answer a various kinds of queries including point and range queries. We propose a new index structure called CD-Tree that can satisfy these requirements.

4.1 The CD-Tree structure

**Definition 3. (CD-Trees)** The CD-Tree of a dataset under a partition is a balanced tree, and has the following properties:

1. It has a root node and  $k+1$  levels, where  $k$  is the dimensionality of dataset.
2. Each dimension of dataset corresponds to a level of the tree. The  $(k+1)$ -level is a special one in which each node records the information of a non-empty cell.
3. Nodes at the  $i$ -th level other than the  $(k+1)$ -th level (non-leaf nodes) contain internal-nodes of the form: (*cNO*, *pointer*), where *cNO* is a keyword, a distinct interval number of  $i$ -th dimension corresponding to a cell. The pointer of  $k$ -th level

internal-nodes points a leaf node. The *pointer* of other level internal-node points to a  $(i+1)$ -th node, which contains all the distinct interval numbers of the next dimension corresponding to the non-empty cells based on intervals of the anterior  $i$  dimensions.

4. A path from the root node to a leaf node corresponds to a cell.

Fig. 2(a) shows the cell structure of a 2-dimension dataset under a partition, and each dimension is divided into 6 intervals, so interval number of each dimension is from 1 to 6. The black cells denote non-empty cells. The cell (2,3) denotes one of which interval number is 2, 3 respectively on these two dimensions. 2 presents the 2<sup>nd</sup> interval on the first dimension, 3 is the 3<sup>rd</sup> interval on the second dimension. Fig. 2(b) is a CD-Tree structure corresponding to the cell structure, which only store non-empty cells. The CD-Tree has 3 levels. The first two levels correspond to  $Y$  dimension and  $Z$  dimension of the dataset respectively, and the last one is cells level. The first dimension has 6 intervals, but only interval 2, 3, 5 and 6 contain data objects, so the root node has 4 internal-nodes. The first internal-node 2 of the root node points to the second level of the CD-Tree, there are two intervals, 2 and 3. A path from the root to a leaf such as (2, 3) corresponds to a cell which stores some information such as data points falling in this cell, the count of points, and etc.

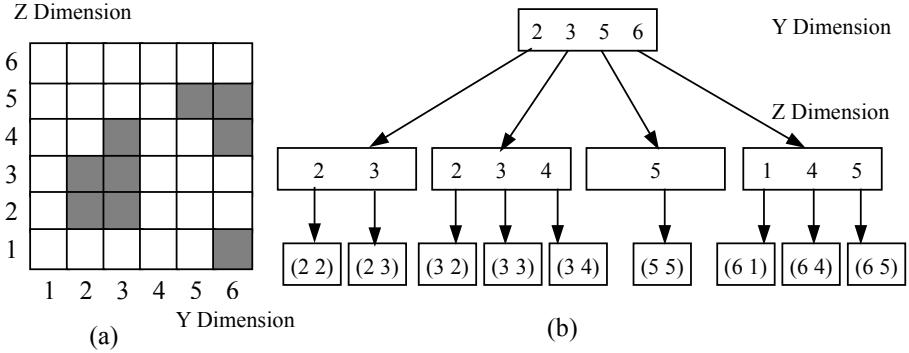


Fig. 2. An example of a CD-Tree

## 4.2 CD-Tree Construction

The algorithm of creating CD-Tree is shown in algorithm 1. Its steps are as follows: step 1 initializes root node of CD-Tree; step 2 reads data objects from database; step 2(a) computes coordinates (i.e. the interval number of each dimension) of each data object; step 2(b) inserts data object into CD-Tree by calling procedure *MapPoint*.

Procedure *MapPoint* starts from first dimension to search coordinate value of a data point in current dimension corresponding to cell in related level of CD-Tree. If all coordinate components of the data object have existed in the CD-Tree, then the information of the data object is stored into the corresponding leaf node of the CD-Tree; if not exist, then create a new node of the next level starting from non-existent level, and create a new leaf node in the final level of the CD-Tree. If current node is

NULL, then create a new node (step 2). If a node exists, there are two cases: first, the value of keyword doesn't exist in the node, then insert it by ascending order (step 3,4). Second, if it exists, then call procedure *MapPoint* to process next level (step 5-7).

**Algorithm 1.** CreateCDTree

**Input:** Dataset  $X$

**Output:** CD-Tree

```

1. Root=NULL;
2. for each data object of  $X$ 
  a. map to CellCoo //CellCoo is the cell's coordinate
  b. MapPoint(CellCoo, point, root, NULL,1)
Procedure MapPoint(CellCoo, point, current, currentPre,
level)
1. if level=dim+1 return; //dim is the dimensionality
2. if Current=NULL then
  a. Current=new Node;
  b. if level=1 then Root=current;
  c. else CurrentPre=current;
3. if DimCoo>maximal keyword of current node then
  // DimCoo is coordinate of current dimension
  a. if level≠dim then create a new interval-node;
  b. else create a new cell, insert the point into the
    cell;
4. else
  a. for each keyword m_Dim in the node
    i. if DimCoo < m_Dim then
      1. insert the keywords into current position;
      2. if level≠dim then create a new pointer for
        next level;
      3. else create a new leaf node, insert point into
        it; goto 5;
    ii. else
      1. if level=dim then
        insert point into the cell; goto 5;
5. currentPre=current;
6. current=current.NextNode;
7. MapPoint(CellCoo,point, current, currentPre,level+1)

```

### 4.3 Range Query on CD-Tree

The range query on a CD-Tree has one cell as center, and the several interval lengths as radius. The results of query are all cells and all points in these cells within the interval lengths.

If there are many keywords in a node, then query efficiency will be affected, so some tricks are used to locate the starting position of query rapidly. As shown in algorithm 2, when the number of keywords less then the upper limit of search range, then locate the starting position of query on the last keyword in the node(step 2), otherwise, locate on the upper limit of query(step 3). This range query scans node from back to front(step 4); in the process of scanning, step 4(a) is used to detect overflow of lower limit; step 4(b) is used to detect overflow of upper limit; step 4(c) calls *RangeQuery* recursively until finding leaf node; step 4(d) stores all position information of leaf node that satisfied query conditions as the query results.

**Algorithm 2.** RangeQuery(*Current*, *CellCoo*, *Radius*, *Level*, *Result*)

**Input:** CD-Tree

*Radius* // the radius of query

*Level* // current level number of CD-Tree

**Output:** *Result*

1. *Size*=number of keywords in current level;  
*DimCoo*=*CellCoo*.*CurrentDimensionKeyword*;
2. **if** (*DimCoo*+*Radius*+1>*Size*) **then**  
*BeginPos*=*Size*-1; //search from the last keyword
3. **else**  
*BeginPos*=*DimCoo*+*Radius*; //search from *DimCoo*+*Radius*
4. **for** (int *i*=*BeginPos*; *i*>-1; *i*-) //from back to front
  - a. **if** ((*DimCoo*-*Radius*)>value of current keyword)  
     **then** 5;
  - b. **else**  
     **if** ((*DimCoo*+*Radius*)<value of current keyword)  
         **then** 4;
  - c. **if** (*Level*≠*Dim*-1) **then**  
     RangeQuery(*Current*.*NextNode*, *CellCoo*, *Radius*, *Level*+1, *Result*);
  - d. **else**  
     *Result*.Add(*Current*, *i*);
5. **return**;

## 5 The Algorithms of Outlier Detection

In [2], the author replaces  $\delta$  with  $M$  where  $M = N \times \delta$ , so the condition that an object  $O$  in a dataset  $X$  is a outlier is as follows: “if no more than  $M$  objects in  $X$  are found in the  $d$ -neighborhood of object  $O$ ”.

In this method, the partition interval is equal to  $d/2\sqrt{k}$ . Each cell has two layers surrounding it. The first layer is one cell thick, while the second is  $\lceil 2\sqrt{k} - 1 \rceil$  cell thick. For a given cell  $w$ , it accumulates three counts – the number of points in the cell, and the number of points in the cell and the first layer together, and the number of points in the cell and both layers together. Let’s refer to these counts as  $Count_w$ ,  $Count_{w2}$  and  $Count_{w3}$ , respectively. The basic idea of the algorithm is as follows: An object  $O$  in the current cell is considered to be an outlier only if  $Count_{w2}$  is less than or equal to  $M$ . If this condition does not hold, then all of the objects in the cell can be removed from further investigation, as they cannot be outliers. If  $Count_{w3}$  is less than or equal to  $M$ , then all of the objects in the cell are considered outliers. Otherwise, if this number is more than  $M$ , then it is possible that some of the objects in the cell may be outliers. To detect these outliers, it is necessary to compare each object  $O$  in the cell with the objects in the second layer. For the objects in the cell, only those having no more than  $M$  points in their  $d$ -neighborhood are outliers. The  $d$ -neighborhood of an object  $O$  is a set of data points that are within distance  $d$  of  $O$ .

We apply the CD-Tree structure to this algorithm. The steps are as follows:

1. Estimate the reasonable division of dataset by pre-computing the SOD under coarse-granularity partitions.

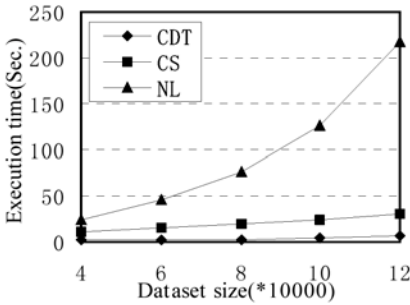
2. Create cells dynamically, insert them into CD-Tree, and map data objects into the cells of CD-Tree.
3. Carry out range searching and outlier detection.

## 6 Experimental Results

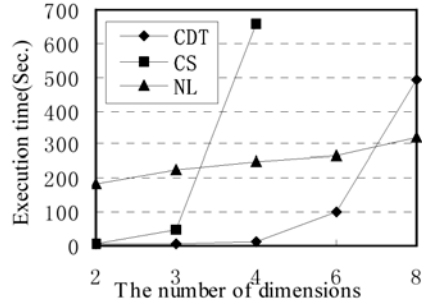
In [2], the authors conducted a number of experiments on historical NHL player data. We repeat their experiments on the NHL96 dataset. We also get the same results under the same parameter.

The following experiments were conducted on Microsoft Windows 2000 Server with 512MB main memory and CPU 2.5GHz. The dataset is image Fourier coefficient dataset, which is widely used in multi-index test (e.g. X-Tree[13]). We use some acronyms to denote different algorithms, as follows: CS is cell-based algorithm in [2]. NL is nested-Loop algorithm implementation in [2]. CDT is the CD-Tree-based algorithm.

We implement these algorithms in the same environment. In [2], the author draws a conclusion that CS is far superior to NL when  $k \leq 4$  dimensions, NL is a wise choice when  $k \geq 5$  dimensions. Thus, we only compare CDT with CS and NL in two cases. First one is the number of dimensions fixed and dataset sizes different, the other one is dataset sizes fixed and the number of dimensions different.



**Fig. 3.** Performance results for different dataset size



**Fig. 4.** Performance results for data dimensionality

Fig.3 shows results for various algorithms and dataset sizes for 3-D,  $\delta=0.999$ . Suppose we change the value of  $d$  to get the same proportion of the points of dataset sizes as outlier. For example, CDT takes 5.624 seconds in total time in a 100000 points dataset. In contrast, CS takes about 30.853 seconds, about 6 times as long, and NL takes longer time 218.04. As Knorr analyzes, the time complexity of NL is  $O(kN^2)$  and CS's is  $O(t_1^k + N)$  where  $k$  is the number of dimensions and  $N$  is the number of points in the dataset. So when the number of dimensions is high, NL outperforms CS. The time complexity of CDT is  $O(t_2^k + N)$ . Because both  $t_1$  and  $t_2$  are relative to the number of cells, and the number of CDT's cells is 1788 which is far less than CS's 195112, then  $t_1 > t_2$ , then leads to the result that CDT outperforms CS.

Fig. 4 gives the execution time of the three algorithms as the number of dimensions is increased from 2 to 8 (the remaining parameters are set to the same values among them). We can find CDT is superior to CS and NL for  $k \leq 7$ , the dimension increases from 4 to 7. The reason is the same as the last instance, the time complexity CS and CDT is  $O(t_1^k + N)$  and  $O(t_2^k + N)$  respectively, dimensions become high, CS is slower than CDT because of  $t_1 > t_2$ .

## 7 Conclusions and Future Work

Outlier detection is an important research issue in data mining. We find the fact that many empty cells exist under a partition when there are data skew in datasets. In order to measure the degree of data skew, we present the concept of SOD. The SOD approximates the percentage of empty cells under a partition, and there exists very large SOD in most of real-life datasets. Based on the concept of SOD, an efficient index structure-CD-Tree is designed, which only stores the non-empty cells. We make some improvements by applying CD-Tree to cell-based algorithm. The time complexity of the cell-based and the CD-Tree-based is  $O(t_1^k + N)$  and  $O(t_2^k + N)$  respectively, but the number of cells in cell-based algorithm is far bigger than CDT's, then  $t_1 > t_2$ . The algorithm based on CD-Tree is far superior to the cell-based algorithm.

There are two directions for our future work. The first one is on how to find local outlier based on CD-Tree, which can be realized by using cluster based on grid [10]. The second one is to further improve the performance of CD-Tree by choosing the suitable order of dimensions to build CD-Tree.

## References

1. D. Hawkins. Identification of Outliers. Chapman and Hall, London, 1980.
2. E. Knorr and R. Ng. Algorithms for Mining Distance-based Outliers in Large Data Sets. VLDB'98(1998)392-430.
3. V. Barnett and T. Lewis. Outliers in Statistical Data. John Wiley and Sons, New York, 1994.
4. E. Knorr and R. Ng. Finding Intensional Knowledge of Distance-based Outliers. VLDB '99(1999)211-222.
5. S. Ramaswamy, R. Rastogi, K. Shim. Efficient Algorithms for Mining Outliers from Large Data Sets. SIGMOD'00(2000)427-438.
6. M. M. Breunig, H-P. Kriegel, R. Ng, J. Sander. LOF: Identifying Density-Based Local Outliers. ACM SIGMOD'00(2000)93-104.
7. C. C. Agarwal and P. Yu. Outlier Detection for High Dimensional Data. SIGMOD' 01.
8. Blake, C. L., Merz, C. J. (1998). UCI Repository of machine learning databases. <http://www.ics.uci.edu/~mllearn/MLRepository.html>. Irvine, CA: University of California, Department of Information and Computer Science.
9. S. Berchtold, D.A. Keim and H. Kriegel. The X-tree: An Index Structure for High-dimensional Data. VLDB'96(1996)28-39.
10. R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. ACM SIGMOD'98(1998)94-105.

# Mobile Data Management with Sound

Ook Lee

College of Information and Communications, Hanyang University, Seoul, Korea  
ooklee@hanyang.ac.kr

**Abstract.** This study introduces a sound-based mobile payment system. Using a mobile phone for payment for mobile commerce is essential for its success. The proposed system will use sound that is generated from the existing mobile phone and the sound can be recognized by the existing credit card reader of the shop with an installation of an ordinary microphone and simple software to process sound input. This system is a better system since it doesn't require customers to buy a new mobile phone and the cost for the microphone and the software is very low.

## 1 Introduction

This article explains a sound-based mobile payment system. Electronic commerce has expanded into commerce with mobile devices, which is called mobile commerce. However in order to implement a truly mobile commerce in the physical world, i.e. purchasing products at a physical shop using a mobile phone, there should be infrastructure elements which can facilitate mobile commerce. In this paper, after presenting the existing mobile phone-based system, a new system that is based on sound of a mobile phone will be discussed as the better solution to promote mobile commerce. There are studies showing the fact that the information technology infrastructure for electronic commerce is a key factor for the success of electronic commerce [1][2]. For example, high-speed Internet service available for most of the nation's population is a critical success factor for the rapid rise of electronic commerce in South Korea. South Korea has more than 90% of mobile phone ownership which provides fertile ground for the growth of electronic commerce as well as mobile commerce. Korea has been recognized worldwide as one of a few nations with highly developed information technology and Internet infrastructure. As for the high-speed Internet usage rate, Korea is reported as the number one country in the world. More than 10 million users subscribe to high-speed Internet services via such means as ADSL, VDSL, community LAN, and satellite. When taking the total number of Korean population into consideration, the high-speed Internet subscription rate in Korea is roughly 21% according to the Korean government ministry news [3]. This is a remarkably high usage rate when compared to those of other OECD countries since the average rate of high-speed Internet subscription in OECD countries is just 1.26% [4]. For many consumers in Korea, the Internet is just a part of their normal daily life. With the high-speed Internet available at very cheap price for almost all households, many electronic commerce firms flourish. Nowadays companies are

selling contents that are downloadable through a mobile phone with a fee, which also has become a highly successful venture. But one of the essential elements of the infrastructure for supporting e-commerce and m-commerce is the payment system, which will be discussed in the next chapter.

## 2 Mobile Phone-Based Payment System for e-Commerce

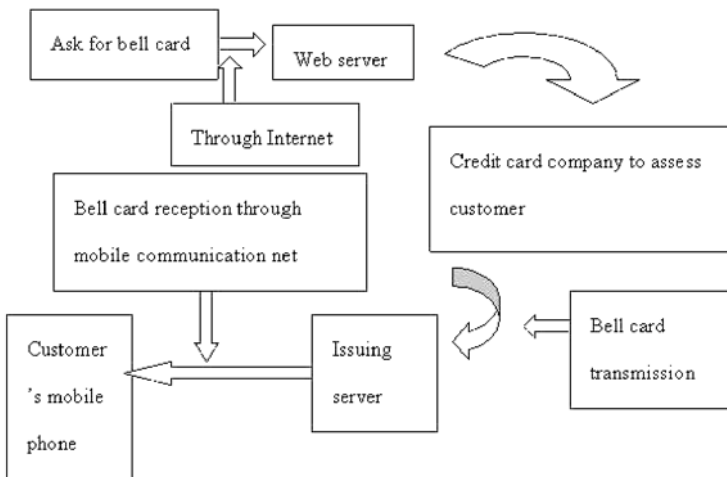
Another key success factor is a viable payment system for electronic commerce. Credit card payment system on the web is widely available and regarded critical for electronic commerce success. However in Korea where credit card usage in general is not as high as countries such as USA, researchers developed many alternative ways to implement a viable payment system. The most successful payment method for web-based electronic commerce is a mobile-phone based one [5]. Customers can type in their mobile phone numbers in a web site which will send the information to the mobile phone company for authorization. When authorization is successful, the customer can purchase products on the web and will take care of the payment through a mobile telephone company's bill which will include purchase records on the web of the customer. This method was very successful in terms of facilitating more participation on web-based electronic commerce especially for people who do not wish to use credit cards for purchase transactions. With the advent of high mobile phone usage and penetration rate, South Korean firms noticed that migrating from fixed location-based electronic commerce to mobile device-based mobile commerce would happen. The mobile phones that are common in South Korea are highly-advanced ones which have capabilities such as web browsing, high-resolution graphics as well as top-quality sound generation leading to high-quality music listening device. People in South Korea can now make purchases on their mobile phone to buy products on the web and download a movie from the Internet and watch from their mobile phones as well as downloading and listening music from mobile phones. But this infrastructure is not enough for facilitating truly mobile commerce. Truly mobile commerce means that the commerce can be processed through the mobile device including the payment in the physical world. In other words, people should be able to make a purchase using a mobile device in a physical shop, not just in the web space. In the web space, as mentioned before, mobile phone-based system which only utilizes software aspect of the each web was widely used. However for the physical world-based mobile commerce, this kind of method is useless. Thus researchers at SK Telecom Corporation developed a new mobile phone as a payment device [6]. This product is called Moneta which refers to a mobile phone with a chip that will generate a unique electronic signal from the mobile phone. This signal represents the customer's purchase identification such as a credit card number. And the shop-keepers need to install a new device to accept the signal from Moneta mobile phone. Although SK Telecom Corporation pushed this method strongly and installed the reading device for many shops, the usage of Moneta mobile phone for purchase transaction has never been improved. The biggest reason for the failure of Moneta as a payment method infrastructure for



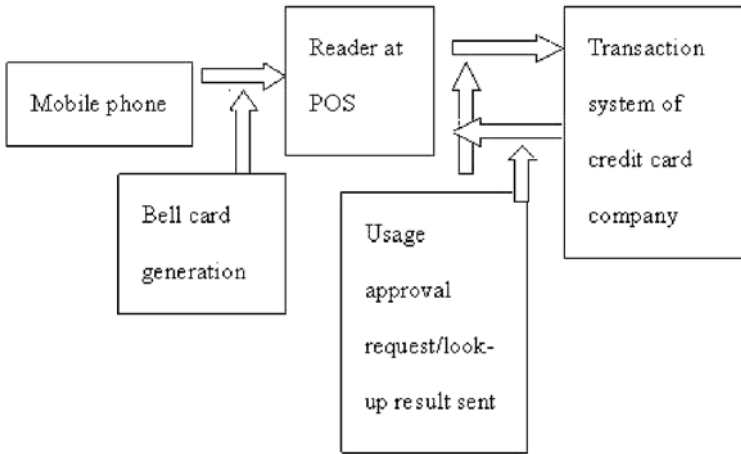
mobile commerce is the fact that people have to buy a new mobile phone in order just to be able to use it for physical mobile transaction at a shop. This is a great barrier since the average price of high-end mobile phone using Moneta is quite high. People do not want to abandon the existing mobile phone which was not cheap either to a more expensive Moneta phone juts for the purpose of obtaining mobile commerce transaction capability.

### 3    Sound-Based Mobile Phone-Based Payment System

Here a new system for physical mobile commerce is introduced. A South Korean firm called Telme Commerce Corporation and the Information System laboratory of Hanyang University developed the sound-based mobile payment system. This system is called Mobile Bell-Card Solution. This solution uses sound-wave for data communication, rather than electronic signal which is used by the existing mobile payment solution, Moneta. The Mobile Bell-Card Solution will use sound that is generated from the existing mobile phone and the sound can be recognized by the existing credit card reader of the shop with an installation of an ordinary microphone and simple software to process sound input. This system is made possible by utilizing the fact that each individual's identification information can be uniquely defined as a sound and the existing card reader with a microphone can receive the sound signal and the signal will be processed by the Point-of Sales computer that is used for card-reading, and processed information will be sent to the credit card company's main computer for authorization. Figure 1 shows how a customer obtains his/her unique bell card. Figure 2 shows how a customer uses the sound-based mobile phone to complete a purchase transaction.



**Fig. 1.** Obtaining a bell card

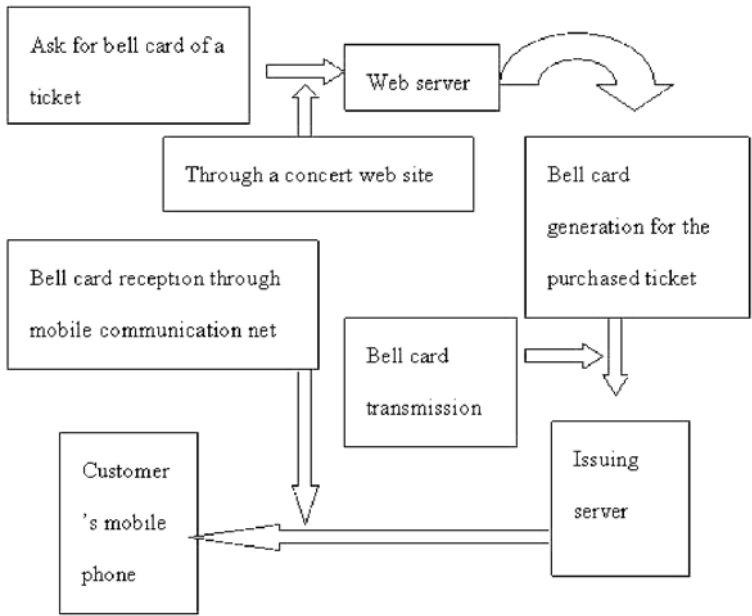


**Fig. 2.** Use of a bell card

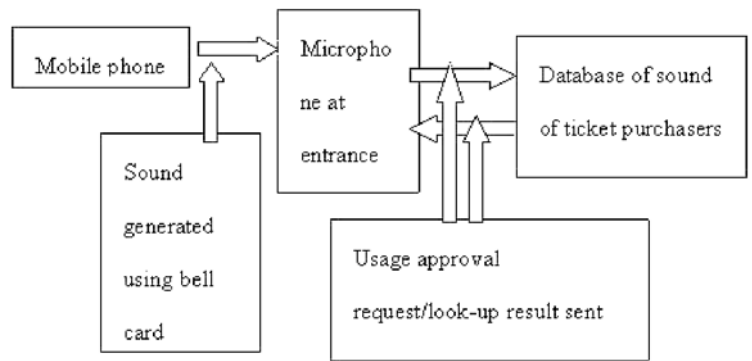
In addition to the application of a bell card for processing at a POS counter, there is another application which is currently used popularly. For many concert goers it can be a hassle to purchase the ticket in advance and submit it later to the attendant. With sound-based mobile data management system, these tickets can be completely eliminated. In other words, one can go to a concert web page where one can purchase the ticket with any kind of payment method including the mobile phone, and a sound that uniquely identifies one's ticket will be sent to the user's mobile phone. After receiving the bell card of the ticket which is a sound file in one's mobile phone, one will go to the concert and at the entrance, activate the sound generation function of the mobile phone which will be absorbed by a microphone at the entrance, and one will be allowed to be inside since the sound will be recognized by a multimedia database that contains records of sound that identifies each purchaser of the ticket. Figure 3 and 4 show this process graphically.

## 4 Conclusion

This system is a better system compared to competing system such as Moneta since it does not require customers to buy a new mobile phone and the shop-keeper has to spend very little for the microphone and the software both of which can be provided free of charge by the credit card company. Eventually when this sound-based payment system is prevalent, bell-card of a mobile phone which only exists in software form will replace physical credit cards. The bell-card will require a PIN number to be activated so that any stolen mobile phone can not be used for unauthorized purchase. The bell-card can also be used for driver's license, passport, etc. due to its ability to define a person's unique identity. This sound-based system is also very marketable since its can be implemented in any mobile phone regardless of its manufacturer. In terms of security, since



**Fig. 3.** Obtaining a bell card from a concert web site



**Fig. 4.** Use of a bell card at a concert

this system uses encryption/description method with a complex cryptography algorithm, this system is copy-protected. At 80DB level sound, 100% of the content of the sound is recognized, which indicates no noise interference. The sound-based mobile payment system must encourage mobile commerce because of its low cost, convenience as well as security.

## References

1. Weill, P. and Vitale, M. Place to Space: Migrating to eBusiness Models. Harvard Business School Press. (2001)
2. Weill, P. and Broadbent, M. Leveraging the New Infrastructure. Harvard Business School Press. (1998)
3. Ministry of Information and Communications, “Korea is the number one in high-speed Internet service”, MIC news, [www.mic.go.kr](http://www.mic.go.kr). (2004)
4. Samsung Economic Research Institute, Home page of SERI, [www.seri.org](http://www.seri.org). (2004)
5. Mobilianz, Inc., Home page of Mobilianz, Inc., [www.mobilianz.co.kr](http://www.mobilianz.co.kr). (2004)
6. Moneta SK Telecommunication Corporation,  
[http://www.sktelecom.com/english/services/m\\_commerce/index.html](http://www.sktelecom.com/english/services/m_commerce/index.html). (2004)

# A Mixture Model of Classical Fuzzy Classifiers

Yongchuan Tang<sup>1,2</sup> and Shouqian Sun<sup>1,2</sup>

<sup>1</sup> College of Computer Science, Zhejiang University  
Hangzhou, Zhejiang Province, 310027, P. R. China  
yongchuan@263.net

<sup>2</sup> State Key Lab of CAD & CG, Zhejiang University  
Hangzhou, Zhejiang Province, 310027, P. R. China  
ssqq@mail.hz.zj.cn

**Abstract.** The classical fuzzy classifier consists of rules each one describing one of the classes. In this paper a new fuzzy classifier is proposed where each rule can represent more than one classes with different probabilities. A learning algorithm based on the gradient descent method is proposed to identify the parameters of the fuzzy sets in the antecedent of each fuzzy rule and the class conditional probabilities associated with each rule. This new approach is applied to the well-known Wisconsin breast cancer classification problem, and a compact, interpretable and accurate fuzzy classifier is achieved.

## 1 Introduction

Fuzzy production rules can deal with the imprecise knowledge and uncertainty information and strengthen the knowledge representation power. Typical fuzzy classifiers consist of interpretable if-then rules with fuzzy antecedents and class labels in the consequent part. The antecedents (if-parts) of the rules partition the input space into a number of fuzzy regions by fuzzy sets, while the consequents (then-parts) describe the output of the classifier in these regions.

In this paper we propose a new fuzzy classification rule, which is different from the classical fuzzy rule. This kind of fuzzy classification rule is similar to that in [1]. That is, the consequent of each fuzzy rule is defined as the probabilities that a given rule represents all classes. The novelty of this new model is that one rule can represent more than one classes with different probabilities. Hence, the same antecedent (fuzzy subregion in the input domain) can infer different classes with different class conditional probabilities.

The paper is organized as follows. Section 2 presents new fuzzy classification rules and its reasoning process. Section 3 describes the identification process based on gradient descent method. Section 4 is our test results in the well-known Wisconsin breast cancer data set. The final section offers the conclusions.

## 2 Structure of the Fuzzy Rule-Based Classifier

The identification of a classifier system means the construction of a model that predicts the class  $y_k = \{c_1, c_2, \dots, c_C\}$  to which pattern  $\mathbf{x}_k = [x_{1k}, x_{2k}, \dots, x_{nk}]$  should be assigned. The classic approach for this problem is based on Bayes' rule,

$$\mathbf{x}_k \text{ is assigned to } c_i \iff p(c_i | \mathbf{x}_k) \geq p(c_j | \mathbf{x}_k) \quad \forall j \neq i. \quad (1)$$

## 2.1 Classical Fuzzy Classifier

Taking into account a classification problem which has  $n$  decision variables (attributes or features)  $X_i, i = 1, 2, \dots, n$  and one class variable  $y$ . Assume  $U_i$  is the universe of discourse of variable  $X_i$ , then the classical fuzzy classification system consists of  $C$  fuzzy if-then rules of the forms:

$$\text{If } (X_1 \text{ is } A_{r1}) \text{ and } (X_2 \text{ is } A_{r2}) \dots \text{ and } (X_n \text{ is } A_{rn}) \text{ then } y \text{ is } c_r \text{ with } \alpha_r; \quad (2)$$

where  $A_{ij}$  is the fuzzy set of  $U_j$  and  $c_r \in \{c_1, c_2, \dots, c_C\}$  is the class label. The parameter  $\alpha_r$  is the certainty factor which means how certain the relationship between the antecedent and the consequent of this rule is. Furthermore, the certainty factor is within the range of  $[0, 1]$ .

Given a case which has  $n$  attribute values  $(x_1, x_2, \dots, x_n)$ , the reasoning method is described as follows:

1. Calculate the degree of activation of each rule. For each rule  $R(r)$  (see (2)),  $A_{r1}(x_1), \dots, A_{rn}(x_n)$  are the membership degrees in the antecedent propositions. The degree of activation of the  $i$ th rule is calculated as:

$$w_r(\mathbf{x}) = \alpha_r \prod_{k=1}^n A_{rk}(x_k). \quad (3)$$

2. Determine the output. The output of the classical fuzzy classifier is determined by the *winner takes all* strategy, i.e.,

$$\hat{y} = c_{i^*}, i^* = \arg \max_{1 \leq i \leq C} w_i(\mathbf{x}). \quad (4)$$

In this paper we employ the gauss-shaped fuzzy sets  $A_{rj}$ , with the following membership function:

$$A_{rj}(x_j) = \exp \left[ -\frac{1}{2} \left( \frac{x_j - a_r^j}{\sigma_r^j} \right)^2 \right], \quad (5)$$

where  $a_r^j$  represents the center and  $\sigma_r^j$  stands for the variance of the Gaussian function.

If we assume that the class conditional probability in the classical fuzzy classifier is defined as follows,

$$p(c_r | \mathbf{x}) = w_r(\mathbf{x}) / \sum_{i=1}^C w_i(\mathbf{x}), \quad (6)$$

then the classical fuzzy classifier is a special case of Bayes classifier.

## 2.2 Fuzzy Classifier with Conditional Probabilities

In this subsection, we propose a new fuzzy classifier. This new fuzzy classifier consists of  $m$  fuzzy classification rule, each fuzzy rule can represents more than one class with different probabilities. That is, the  $r$ th fuzzy classification rule has the form as follows,

$$\begin{aligned} &\text{If } (X_1 \text{ is } A_{r1}) \text{ and } (X_2 \text{ is } A_{r2}) \dots \text{ and } (X_n \text{ is } A_{rn}) \\ &\text{then } y \text{ is } c_1 \text{ with } p(c_1 | R_r) \dots y \text{ is } c_C \text{ with } p(c_C | R_r) \end{aligned} \quad (7)$$

satisfying  $0 \leq p(c_i | R_r) \leq 1, \sum_{i=1}^C p(c_i | R_r) = 1$ .

The above formulation shows that the same antecedent can infer different classes with different probabilities. In the classical fuzzy model, the same antecedent can just infer one definite class with some degree of certainty. This new fuzzy model is actual a mixture of classical fuzzy classifiers since the Bayes decision rule in new fuzzy classifier is defined as follows,

$$p(c_r | \mathbf{x}) = \sum_{i=1}^m p(R_i | \mathbf{x}) p(c_r | R_i), \quad (8)$$

where  $p(R_i | \mathbf{x}) = A_i(\mathbf{x}) / \sum_{i=1}^m A_i(\mathbf{x})$ .

### 3 Identification of Fuzzy Classifier

Assume that the training data set  $D$  consists of  $N$  training cases  $[\mathbf{x}_j; y_j] = [x_{1j}, x_{2j}, \dots, x_{nj}; y_j]$ , where  $y_j \in \{c_1, c_2, \dots, c_C\}$ ,  $j = 1, \dots, N$ . The object to identify the parameters in the fuzzy classifier is to find the parameters to best model the data set. Since we make the classification decision by maximizing posteriori class probability, the notion of “best” means that maximizing the objection function

$$Q(D) = \prod_{j=1}^N p(y_j | \mathbf{x}_j). \quad (9)$$

We view  $Q(D)$  as the function of the parameters in the fuzzy classifier. In fact, it turns out to be easy to maximize the log-function  $\ln Q(D)$ . Since two functions are monotonically related, maximizing one is equivalent to the other. By Jensen inequality, we have

$$\ln Q(D) \geq \sum_{j=1}^N \sum_{i=1}^m \mu_{ij} \ln P(y_j | R_i) = H(D), \quad (10)$$

where

$$\mu_{ij} = A_i(\mathbf{x}_j) / \sum_{i=1}^m A_i(\mathbf{x}_j). \quad (11)$$

Since log-function  $\ln Q(D)$  is lower bounded by function  $H(D)$ , we can maximize  $H(D)$ . In fact, this maximization problem is a constrained optimization problem. That is, for every  $i$ , we must have that  $\sum_{h=1}^C p(c_h | R_i) = 1$ . Introducing Lagrange multipliers for these constraints, we have the following formula,

$$p(c_k | R_i) = \left( \sum_{j: y_j = c_k} \mu_{ij} \right) / \sum_{j=1}^N \mu_{ij}, \quad (12)$$

for all  $i, k$ .

In order to determine the shapes of fuzzy sets in the antecedent of each fuzzy rule, we adopt the gradient-based method to resolve this problem.

The partial derivative with respect to the parameter  $a_r^k$  can be obtained as follows,

$$\frac{\partial H(D)}{\partial a_r^k} = \sum_{j=1}^N \frac{x_{kj} - a_r^k}{(\sigma_r^k)^2} \mu_{rk} \left[ \ln p(y_j | R_r) - \sum_{h=1}^m \mu_{hj} \ln p(y_j | R_h) \right] \quad (13)$$

Similarly, we can obtain the partial derivative with respect to the parameter  $\sigma_r^k$ ,

$$\frac{\partial H(D)}{\partial \sigma_r^k} = \sum_{j=1}^N \frac{(x_{kj} - a_r^k)^2}{(\sigma_r^k)^3} \mu_{rk} \left[ \ln p(y_j | R_r) - \sum_{h=1}^m \mu_{hj} \ln p(y_j | R_h) \right] \quad (14)$$

We can now summarize the above discussion in the form of a basic algorithm for learning the fuzzy classifier with conditional probabilities from the training data set. For sake of clarity, we show this basic algorithm in the Fig. 1.

---

```

function F-Algorithm(F,  $D$ ) returns an optimized fuzzy classifier
  inputs: F, a fuzzy classifier with parameters  $\mathbf{w} = (a_r^k, \sigma_r^k)$ ;  $D$ , training examples
  repeat until  $\Delta \mathbf{w} \approx 0$ 
    for each class  $c_k$  and the  $r$ th rule
      compute probability  $p(c_k | R_r)$  in F according to the Eq. (12)
    for each variable  $X_k$ , the  $i$ th rule
      compute  $\frac{\partial H(D)}{\partial a_r^k}$  according to the Eq. (13), compute  $\frac{\partial H(D)}{\partial \sigma_r^k}$  according to the Eq. (14)
       $a_r^k \leftarrow a_r^k + \alpha \frac{\partial H(D)}{\partial a_r^k}$ ,  $\sigma_r^k \leftarrow \sigma_r^k + \alpha \frac{\partial H(D)}{\partial \sigma_r^k}$ ,  $\Delta \mathbf{w} = (\frac{\partial H(D)}{\partial a_r^k}, \frac{\partial H(D)}{\partial \sigma_r^k})$ ,  $\mathbf{w} = (a_r^k, \sigma_r^k)$ 
  return F

```

---

**Fig. 1.** A basic algorithm.

In order to initialize the parameters of the fuzzy classifier, we generate a random matrix  $U$  such that

$$0 < \mu_{rj} < 1, \sum_{r=1}^m u_{rj} = 1, r = 1, \dots, m, j = 1, \dots, N \quad (15)$$

then we use the following formulas to estimate the initial mean values and widths of gaussian membership functions in the fuzzy classifier,

$$a_r^k = \frac{\sum_{j=1}^N \mu_{rj} x_{kj}}{\sum_{j=1}^N \mu_{rj}}, \quad (\sigma_r^k)^2 = \frac{\sum_{j=1}^N \mu_{rj} (x_{kj} - a_r^k)^2}{\sum_{j=1}^N \mu_{rj}}. \quad (16)$$

## 4 Performance Evaluation

Wisconsin breast cancer data is widely used to test the effectiveness of classification and rule extraction algorithms. The aim of the classification is to distinguish between benign



and malignant cancers based on the available nine measurements. The measurements are assigned an integer value between 1 and 10, with 1 being the closest to benign and 10 the most anaplastic. In the original database, the class distribution is 65.5% benign and 34.5% malignant, respectively.

**Table 1.** The classification results of new fuzzy classifier.

Number of rules	Min Acc.	Max Acc.	Mean Acc.
$m = 2$	95.59%	100%	97.16%
$m = 3$	98.53%	100%	98.56%

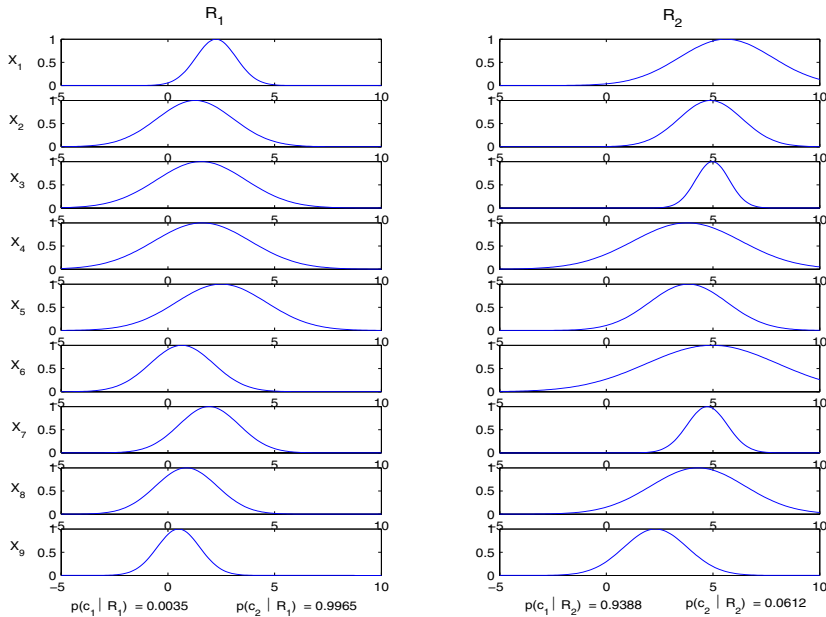
Partial previous work on the classification problem for the Wisconsin breast cancer is listed in the reference [1]. The advanced version of C4.5 gives misclassification of 5.26% on 10-fold cross validation with tree size  $25 \pm 0.5$  [2]. [3] use the decision tree to initialize the fuzzy classifier and GAs to improve the classification accuracy. The classification accuracy of this fuzzy system with two rules on the Wisconsin breast cancer data is 96.87%. [4] developed a constrained-syntax genetic programming system for discovering classification rules in the Wisconsin breast cancer data, the classification accuracy is compatible with the advanced version of C4.5[2]. [5] combined neuro-fuzzy techniques with interactive strategies for rule pruning to obtain a fuzzy classifier. The final fuzzy classifier for the Wisconsin breast cancer could be reduced to two rules with 5 - 6 features only, with a misclassification of 4.94% on 10-fold validation.

The performance of the obtained classifier measured by 10-fold cross validation. In Table 1, we list the experimental results on the Wisconsin breast cancer database by using the fuzzy classifier with conditional probabilities. This table shows that our fuzzy classifier can achieve very high accuracy with simple model structure. The fuzzy classifier with 2 fuzzy classification rules achieves 97.16% mean accuracy, and the fuzzy classifier with 3 fuzzy classification rules achieves 98.56% mean accuracy. In this experiment, the learning rate  $\alpha = 0.001$  and iteration number is 1000.

Figure 2 illustrates the corresponding final rules of the fuzzy classifier trained by gradient-based method. From this figure, we find that the centers of the gaussian memberships in the first rule are on the left of the corresponding universes, and that in the second rule are on the right of the corresponding universes.

## 5 Conclusions

In this paper we provide a gradient-based method to identify a new fuzzy classification system from observation data. This fuzzy classifier consists of rules each one represents more than one class with different conditional probabilities. This fuzzy classifier can be represented as a mixture model of classical fuzzy classifiers. A compact and interpretable fuzzy classifier is achieved by applying the proposed method to the well-known Wisconsin breast cancer data. The test results show that this fuzzy classification system has strong power of knowledge representation.



**Fig. 2.** The final fuzzy rules in fuzzy classifier with conditional probabilities.

## Acknowledgements

This work has been supported by China Postdoctoral Science Foundation, Hubei province young elitist project (No. 2002AC001) and Chinese 973 project (No. 2002CB312106).

## References

1. Abonyi, J., Szeifert, F.: Supervised fuzzy clustering for the identification of fuzzy classifiers. *Pattern Recognition Letters* **24** (2003) 2195–2207
2. Quinlan, J.R.: Improved use of continuous attributes in c4.5. *J. Artificial Intell. Res.* **4** (1996) 77–90
3. Janos Abonyi, J.A., Szeifert, F.: Data-driven generation of compact, accurate, and linguistically sound fuzzy classifiers based on a decision-tree initialization. *International Journal of Approximate Reasoning* **32** (2003) 1–21
4. Celia C. Bojarczuk, Heitor S. Lopes, A.A.F., Michalkiewicz, E.L.: A constrained-syntax genetic programming system for discovering classification rules: application to medical data sets. *Artificial Intelligence in Medicine* **30** (2004) 27–48
5. Nauck, D., Kruse, R.: Obtaining interpretable fuzzy classification rules from medical data. *Artificial Intell. Med.* **16** (1999) 149–169

# An Empirical Study of Building Compact Ensembles

Huan Liu, Amit Mandvikar, and Jigar Mody

Computer Science & Engineering  
Arizona State University  
Tempe, AZ 85281

{huan.liu, amitm, jigar.mody}@asu.edu

**Abstract.** Ensemble methods can achieve excellent performance relying on member classifiers' accuracy and diversity. We conduct an empirical study of the relationship of ensemble sizes with ensemble accuracy and diversity, respectively. Experiments with benchmark data sets show that it is feasible to keep a small ensemble while maintaining accuracy and diversity similar to those of a full ensemble. We propose a heuristic method that can effectively select member classifiers to form a compact ensemble. The idea of compact ensembles is motivated to use them for effective active learning in tasks of classification of unlabeled data.

**Keywords:** Image Mining, Ensemble Methods, Compact Ensemble

## 1 Introduction

Many Applications generate massive unlabeled data in forms of text, image, audio or multimedia. A commonly seen task in real-world applications is *classification*. Using an image classification task as an example, a user may be able to carefully study a few images at a time, but may not have the patience and consistent performance to hand-label a hundred of training images. To make things worse, new images are being collected and waiting to be classified. A real-world problem we face is to classify Egerai Densa in images. Egeria is an exotic submerged aquatic weed causing navigation and reservoir-pumping problems in the west coast of the USA. As a part of a control program to manage Egeria, classification of Egeria regions in aerial images is required. This task can be stated more specifically as one of classifying massive data *without class labels*. Relying on human experts for labeling Egeria regions is not only time-consuming and costly, but also inconsistent in their performance of labeling. Massive manual classification becomes impractical when images are complex with many different objects (e.g., water, land, Egeria) under varying picture-taking conditions (e.g., deep water, sun glint). In order to automate Egeria classification, we need to ask experts to label images, but want to minimize the task.

Many data mining methods for classification are available to help massively process image data. In order for classification methods to work, labeled data is needed for training purpose. We face a dilemma: to classify unlabeled data, we need to rely on a classification algorithm; in order for the classification algorithm to learn from the training data, we need to have data labeled. Since we have to have labeled data for training,

we ask if it is possible that the classification algorithm can learn with as few labeled data as possible. By doing so, we can minimize the labeling efforts by experts to turn unlabeled data to labeled one. Active learning [5] is an effective learning framework that can be applied in the above process of working with domain experts and using as few labeled data as possible and learn to perform classification through an iterative (re)learning process. Active learning requires highly accurate classifiers that ideally can generalize well with a small set of labeled data. Therefore, we examine one type of highly accurate classifiers - ensemble methods. We analyze one ensemble method (Bagging [2]) with experiments on benchmark data sets, observe interesting results from the experiments, and evaluate its feasibility and effectiveness to use compact ensembles for active learning.

## 2 Ensemble Methods

Ensemble methods are learning algorithms that construct a set of classifiers and then classify new instances by combining the individual predictions. An ensemble often has smaller expected loss or error rate than any of the  $n$  individual (member) classifiers [7]. A good ensemble is one whose members are both *accurate* and *diverse* [3].

An accurate classifier is one that has an error rate of better than random guessing on new instances; more specifically, each member classifier should have its error rate below 0.5. Two classifiers are diverse if they make different (or uncorrelated) errors on new data points. In reality, the errors made by member classifiers will never be completely independent of each other, unless the predictions themselves are completely random (in which case the error rate will be greater than 0.5) [3]. However, so long as each member's error rate is below 0.5, with a sufficient number of members in an ensemble making somewhat uncorrelated errors, the ensemble's error rate can be very small as a result of voting. Out of the many proposed ensemble methods, we consider Bagging in this work as it is the most straightforward way of manipulating the training data [3]. Bagging relies on bootstrap replicates of the original training data to generate multiple classifiers that form an ensemble. Each bootstrap replicate contains, on the average, 63.2% of the original data, with several instances appearing multiple times.

Ensembles can be assessed by three measures: predictive accuracy, diversity, and size. Accuracy can be estimated using cross validation. Diversity measures how different the predictions member classifiers made in an ensemble. The first two measures are of the goodness of an ensemble. The last one is about ensemble size - the number of member classifiers required for ensemble learning. Ensemble size mainly hinges on the complexity of the training data. For a fixed type of classifier (say, decision trees), the more complex the underlying function of the data is, the more members an ensemble needs. The complexity of the function can always be compensated by increasing the number of members for a given type of classifier until the error rate converges.

Following [4], let  $\hat{Y}(x) = \hat{y}_1(x), \dots, \hat{y}_n(x)$  the set of the predictions made by member classifiers  $c_1, \dots, c_n$  of ensemble  $C$  on instance  $\langle x, y \rangle$  where  $x$  is input, and  $y$  is prediction. The **ensemble prediction** of a uniform voting ensemble for input  $x$  under loss function  $l$  is  $\hat{y}(x) = \arg \min_{y \in Y} E_{c \in C} [l(y, \hat{y}_c(x))]$ . The ensemble prediction is the one that minimizes the expected loss between the ensemble prediction and the predic-

tions made by each member classifier  $c$  for the instance  $\langle x, y \rangle$ . The **loss** of an ensemble on instance  $\langle x, y \rangle$  under loss function  $l$  is given by  $L(\langle x, y \rangle) = l(\hat{y}(x), y)$ . The error rate of a data set with  $N$  instances can be calculated as  $e = \frac{1}{N} \sum_{i=1}^N L_i$  where  $L_i$  is the loss for instance  $x_i$ . **Accuracy** of an ensemble is  $1 - e^1$ . The **diversity** of an ensemble on input  $x$  under loss function  $l$  is given by  $\overline{D} = E_{c \in C} [l(\hat{y}_c(x), \hat{y}(x))]$ .

The diversity is the expected loss incurred by the predictions of the member classifiers relative to the ensemble prediction. Commonly used loss functions include square loss ( $l_2(\hat{y}, y) = (\hat{y} - y)^2$ ), absolute loss ( $l_1(\hat{y}, y) = |\hat{y} - y|$ ), and zero-one loss ( $l_{01}(\hat{y}, y) = 0$  iff  $\hat{y} = y$ ;  $l_{01}(\hat{y}, y) = 1$  otherwise). We use zero-one loss in this work. We conduct experiments below.

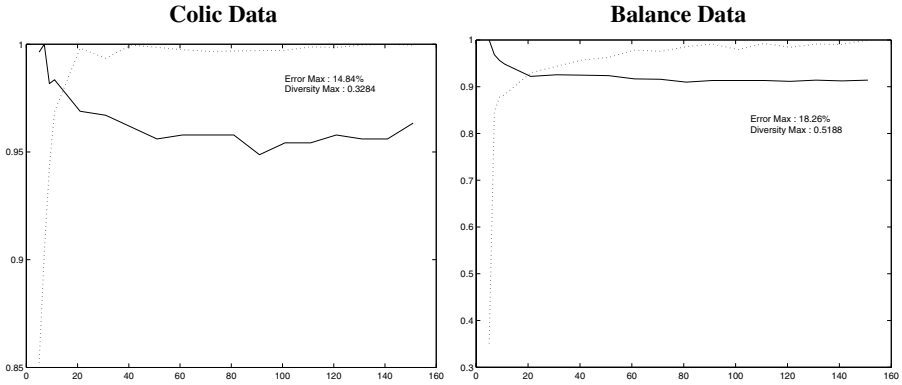


Fig. 1. Plots for normalized diversity and error rates.

Having ensemble loss (or accuracy) and diversity defined, we investigate how ensemble sizes influence ensemble accuracy and diversity. We use benchmark data sets [1] in the experiments. These data sets have different numbers of classes, different types of attributes and are from different application domains.

We use the Weka [6] implementation of Bagging [2] as the ensemble generation method and used J4.8 (the Weka's implementation of C4.5 without pruning as the base learning algorithm. For each data set, we run 10-fold cross validation of Bagging and increase ensemble sizes from 5 to 151 and record each ensemble's error rate  $e$  and diversity  $D$ . Their average values  $\bar{e}$  and  $\bar{D}$  are calculated.

We have run experiments with 18 ensemble sizes (5, 7, 9, 11, 21, 31, 41, 51, 61, 71, 81, 91, 101, 111, 121, 131, 141, and 151) with 10-fold cross validation for each data set (29 sets in total). In Figure 1, two illustrative sets of curves are demonstrated. Both diversity values (**dashed lines**) and error rates (**solid lines**) are normalized for plotting purposes. The vertical axis shows percentage ( $p$ ). The maximum values of diversity and error rate are given in each figure. We can derive absolute values for diversity and error rates following  $Max \times p$ . The trends of diversity and error rates are of our interest. We

<sup>1</sup> We use *loss* and *error* interchangeably in this paper.

can observe a general trend that diversity values increase and approach to the maximum, and error rates decrease and become stable as ensemble size increases.

The results of this study prompt us to think that if we can find the ensemble with minimum size for every application while maintaining accuracy and diversity, we will be able to make the retraining of the ensemble very fast. One way of searching for such ensembles is what we did in our experiments: increasing ensemble sizes until the curves of error rates and diversity stabilize. However, this is a very costly process when ensemble sizes are large. For example, to estimate error rate and diversity of an ensemble with 100 member classifiers using 10-fold cross validation, we need to build  $100 \times 10$  classifiers. The total number of classifiers required to build starting an ensemble with 5 classifiers is  $(5 + 6 + \dots + 100) \times 10 = 50,400$  with each classifier taking  $O(N \log N)$  time to train, where  $N$  is the number of instances of a training data set. There is a need for an alternative that can determine ensemble size without training so many classifiers.

### 3 Compact Ensembles via Classifier Selection

In general, 50-100 member classifiers have been used to build ensembles. In the context of active learning, since the initial training of ensembles is off-line, it is feasible to have an ensemble with 100 member classifiers. To generate a training set for the task of selecting member classifiers, we first perform Bagging with 100 member classifiers. We then use the learned classifiers ( $c_k$ ) to generate predictions for instance  $\langle x_i, y_i \rangle$ :  $\hat{y}_i^k = c_k(x_i)$ . The resulting data set consists of instances of the form  $((\hat{y}_i^1, \dots, \hat{y}_i^M), y_i)$ . After this data set is constructed, the problem of selecting member classifiers becomes one of feature selection.

As discussed before, when member classifiers are equally good, the key issue to find a subset of classifiers that maintain diversity. When we have the newly formed training data, selecting a subset of diverse classifiers is equivalent to selecting a subset of features using diversity of the subset as the goodness criteria. In order to build dual ensembles, we divide data  $D_{new}$  into two data sets  $D_{new}^1$  and  $D_{new}^0$  according to the class labels. For each data set, we can calculate the full ensemble's diversity, then look for the smallest subset of classifiers that can maintain this diversity. This will result in two ensembles  $E_1$  for  $D_{new}^1$  and  $E_0$  for  $D_{new}^0$ .

---

#### Selecting Diverse Classifiers

1. Divide data  $D_{new}$  according to its last column  $y_j$  form data  $D_{new}^1$  and  $D_{new}^0$  for classes 1 and 0;
  2. For data set  $D_{new}^1$ 
    - Calculate diversity  $D_{full}$  for ensemble  $E_{full}$ ;
    - $S_{best} = S_{full}$ ;
    - $D_{best} = D_{full}$ ;
    - Apply  $LVF_d$  with  $D_{full}$  as a goodness criterion:
      - $E_{temp}$  is generated by  $LVF_d$ ;
      - Calculate  $D_{temp}$  and  $S_{temp}$  for  $E_{temp}$
      - If  $D_{temp} \approx D_{full} \wedge S_{temp} < S_{best}$ 
        - $D_{best} = D_{temp}$ ;
        - $S_{best} = S_{temp}$ ;
  3. Repeat step 2 for data  $D_{new}^0$
- 

**Fig. 2.** Searching for small ensembles.

We implement a modified version of  $LVF$  - a filter model of feature selection algorithm (its implementation can be found in Weka [6]) and we call it  $LVFd$  as it uses diversity instead of consistency as the goodness measure of feature subsets. The basic idea is as follows: randomly generate a subset of classifiers as  $E_{temp}$ ; calculate the diversity  $D_{temp}$  and size  $S_{temp}$  of  $E_{temp}$ ; if  $D_{temp}$  is similar to the diversity  $D_{full}$  of the full ensemble  $C$ ,  $E_{temp}$ 's diversity and size are remembered as  $D_{best}$  and  $S_{best}$ ; in the subsequent steps, only if a new  $E_{temp}$ 's diversity  $D_{temp}$  is similar to  $D_{full}$  and size  $S_{temp}$  is smaller than  $S_{best}$ ,  $D_{best}$  and  $S_{best}$  with  $E_{best}$  are updated with those of  $E_{temp}$ .  $LVFd$  stops when  $S_{best}$  does not change after a given number of iterations. The algorithm of  $LVFd$  is given in Figure 2 in which  $\approx$  means "is similar to". Similarity can be defined by the measures for two comparing ensembles. In our implementation, we define  $p \geq 1$  as a threshold for similarity definition: if the difference between the two measures is less than  $p\%$ , we consider they are similar.  $p$  is set as 1 in our experiments.

**Table 1.** Selected Compact vs. Full Ensembles for training data.

Dataset	$E_{full}$			$E_s$					Acc Diff
	Accuracy	$DivE1$	$DivE0$	Accuracy	$DivE1$	$DivE0$	$SizeE1$	$SizeE0$	
breast	98.43 $\pm$ 0.12	0.04	0.18	97.00 $\pm$ 0.51	0.04	0.02	8	13	-1.43
breast-c	83.21 $\pm$ 1.72	0.21	0.06	79.72 $\pm$ 1.53	0.22	0.07	17.33	7	-3.50
colic	87.64 $\pm$ 1.05	0.08	0.04	87.77 $\pm$ 1.15	0.09	0.05	26.33	11	0.13
credit-a	93.62 $\pm$ 0.24	0.07	0.08	91.74 $\pm$ 0.51	0.07	0.09	18.33	5	-1.88
credit-g	92.90 $\pm$ 0.64	0.27	0.10	88.30 $\pm$ 0.26	0.26	0.10	19	10	-4.60
diabetes	95.31 $\pm$ 0.16	0.21	0.10	90.36 $\pm$ 0.56	0.21	0.11	18.33	11	-4.95
heart-st	96.67 $\pm$ 0	0.13	0.10	92.78 $\pm$ 2.00	0.94	0.11	18.33	10	-3.89
hepatits	94.84 $\pm$ 0.52	0.06	0.26	89.36 $\pm$ 1.64	0.08	0.28	8.33	19	-5.48
ionosphr	99.72 $\pm$ 0.12	0.03	0.09	95.58 $\pm$ 1.37	0.04	0.10	3.67	13	-4.13
kr	99.56 $\pm$ 0.05	0.01	0.01	99.23 $\pm$ 0.16	0.01	0.01	6.33	4	-0.33
labor	73.68 $\pm$ 8.13	0.09	0.12	92.11 $\pm$ 0	0.11	0.14	7.67	22	18.42
vote	97.36 $\pm$ 0.25	0.02	0.02	96.78 $\pm$ 0.47	0.03	0.02	8.33	11	-0.58

## 4 Experiments

We now conduct some further study on the benchmark data sets and evaluate if we can apply the algorithm in Figure 2 to find compact ensembles with comparable performance of full ensembles.

Among the benchmark data sets used in Section 2, we use those with binary classes for further experiments using dual ensembles - building one ensemble for each class. As we discussed earlier, an ensemble with 100% accurate member classifiers has diversity value 0. Hence, this data set is excluded. 3-fold cross validation is conducted in this experiment. We record the results in the training and testing phases. For the training data, we record diversity values for  $E_{full}^1$  and  $E_{full}^0$  and for  $E_s^1$  and  $E_s^0$  as well as their accuracy rates in Table in Figure 3. We also record the average ensemble sizes for  $E_s^1$  and  $E_s^0$ .

**Table 2.** Selected Compact vs. Full Ensemble for testing data.

Dataset	$E_{full}$	$E_s$	Acc Diff
	Accuracy	Accuracy	
breast	96.57 $\pm$ 0.73	95.85 $\pm$ 1.04	-0.72
breast-c	68.51 $\pm$ 2.70	65.71 $\pm$ 2.62	-2.80
colic	85.59 $\pm$ 1.61	85.05 $\pm$ 1.47	-0.54
credit-a	87.39 $\pm$ 0.36	83.04 $\pm$ 0.71	-4.35
credit-g	75.60 $\pm$ 0.85	69.80 $\pm$ 0.75	-5.80
diabetes	75.26 $\pm$ 1.29	69.01 $\pm$ 2.30	-6.25
heart-st	80.74 $\pm$ 1.32	74.81 $\pm$ 1.68	-5.93
hepatitis	80.67 $\pm$ 1.74	76.80 $\pm$ 2.33	-3.87
ionosphr	92.02 $\pm$ 1.86	86.32 $\pm$ 1.85	-5.70
kr	99.28 $\pm$ 0.09	98.78 $\pm$ 0.16	-0.50
labor	77.19 $\pm$ 3.79	84.21 $\pm$ 0	7.02
vote	95.40 $\pm$ 0.82	94.94 $\pm$ 0.50	-0.46

We observe that (1) **Diversity** – Selected ensembles can maintain similar or higher diversity than those of full ensembles (Table 1) on the training data. (2) **Ensemble size** – Average sizes for  $E_1$  and  $E_0$  are 13.33 and 11.33, the size difference between selected ensembles and full ensembles is about 75 (the reduction is significant). (3) **Accuracy** – Selected ensembles have lower accuracy than that of the full ensemble on both training and testing data (on average, 1.01% and 2.49% less, respectively). Along with the reduction in ensemble size, it is reasonable.

## 5 Conclusions

Classification of unlabeled data is a common task in real-world applications. We discuss an application of classifying unlabeled images for the purpose of detecting Egeria. Motivated by this application and aiming to alleviate the burden of experts to manually label numerous images, we propose to employ active learning. We analyze what is required to apply active learning and conclude that highly accurate ensemble methods can be used as base classifiers for active learning with class-specific ensembles (e.g., dual ensembles for a binary class problem). As active learning is an iterative process, it requires that each base classifier is efficient to train and test. This directly translates to the necessity of using compact ensembles. We suggest that ensemble size can be reduced as long as its diversity is maintained. A classifier selection algorithm is proposed to find the smallest ensemble that maintains similar accuracy. Various experiments are conducted using benchmark data sets for demonstration and validation purposes.

## Acknowledgments

This work is in part sponsored by CEINT ASU 2004.

## References

1. C. Blake and C. Merz. UCI repository of machine learning databases, 1998.
2. L. Breiman. Bagging predictors.
3. T. Dietterich. Ensemble methods in machine learning. In *First International Workshop on Multiple Classifier Systems*, pages 1–15. Springer-Verlag, 2000.
4. M. Goebel, P. Riddle, and M. Barley. A unified decomposition of ensemble loss for predicting ensemble performance. In *Proceedings of the 19th ICML*, pages 211–218. 2002.
5. G. Schohn and D. Cohn. Less is more: Active learning with support vector machines. In *Proceedings of the 17th ICML*, pages 839–846, 2000.
6. I. Witten and E. Frank. *Data Mining - Practical Machine Learning Tools and Techniques with JAVA Implementations*. Morgan Kaufmann Publishers, 2000.
7. Z.-H. Zhou, Y. Jiang, and S.-F. Chen. Extracting symbolic rules from trained neural network ensembles. *AI Commun.*, 16(1):3–15, 2003.



# Merging Individually Learned Optimal Results to Accelerate Coordination

Huaxiang Zhang and Shangteng Huang

Department of Computer Science and Technology, Shanghai Jiaotong University  
200030 Shanghai, P. R. China  
Zhxx@sjtu.edu.cn

**Abstract.** By merging agents' individually learned optimal value functions, agents can learn their optimal policies in a multiagent system. Pre-knowledge of the task is used to decompose it into several subtasks and this decomposition greatly reduces the state and action spaces. The optimal value functions of each subtask are learned by MAXQ-Q [1] algorithm. By defining the lower and upper bound on the value functions of the whole task, we propose a novel online multiagent learning algorithm LU-Q, and LU-Q accelerates learning of coordination between multiple agents by task decomposition and action pruning.

## 1 Introduction

Learning of coordination between multiagent is intractable. If the whole task can be decomposed into several subtasks and all agents have already learned subtasks individually, this learned knowledge should be useful for all agents to find the optimal policies of the whole task. By adopting MAXQ task decomposition method proposed by Dittmann [1], we propose a theoretical-sound multiagent learning algorithm LU-Q which has no constraint on subtask's value functions.

The algorithm proposed by Singh et al [3] requires all state values of each subtask be positive and the model of each subtask be known. Ghavamzadeh et al's algorithm [4] requires state-action values be positive too. These two algorithms didn't take task decomposition into account, so the application of them is quite limited.

We use MAXQ-Q to learn the optimal value functions of each subtask and transform the learned optimal value functions of each subtask to loosen the constraints on the value functions. The analysis and experimental results show LU-Q learns more efficiently than standard Q learning [2].

## 2 Multiagent Markov Decision Processes(MMDPs)

We use Markov decision processes (MDPs) as the problem description framework. A MDP is a 4-tuple  $\langle S, A, P, R \rangle$ , and each term represents the state set, the action set,

the transition probability and the reward function respectively.  $P: S \times A \times S \rightarrow [0,1]$  and  $p(s' | s, a)$  refers to the probability from state  $s$  to  $s'$  when action  $a$  is executed.  $R: S \times A \rightarrow \mathfrak{R}$  and  $r(s' | s, a)$  is the immediate reward when  $a$  is executed in  $s$  and transits to  $s'$ . A policy  $\pi$  is a function mapping from a state to an action. The value function  $Q^\pi(a, s)$  is obtained by executing  $a$  in  $s$  and then following  $\pi$  afterward. An optimal policy  $\pi^*$  maximizes  $Q^\pi(a, s)$  and we denote the optimal value as  $Q^*(a, s)$  and call it the optimal value function.

A MMDP [5] can be described as a combination of multiple MDPs. We use a 5-tuple  $\langle M, S, A, P, R \rangle$  to represent it.  $M$  is the set of agents, and the other 4 terms refer to joint state set, joint action set, joint probability and joint reward respectively.  $S \in S_1 \times \dots \times S_k$  is the state set ( $S_i$  is the state set of agent  $i$ , and  $k$  the number of agents), and  $A \in A_1 \times \dots \times A_k$  is the joint action set ( $A_i$  is the action set of  $i$ ).  $p(s' | a, s) = \prod_{i=1}^k p(s'_i | a_i, s_i)$ ,  $r(s' | s, a) = \sum_{i=1}^k r(s'_i | s_i, a_i)$ . By defining the value functions of multiple MDPs, the joint optimal policy can be learned.

### 3 Learning with Upper and Lower Bound

When the model of the task is unknown, online learning is needed. Suppose a collection of  $k$  agents has learned their optimal policies for each subtask, we now need to find their joint optimal policies for the whole task. This joint optimal solution is the optimal policy of the MMDP. If there is no interaction between the agents, it's clear the profile of optimal policies for each subtask is the solution to the whole task. In real problems, actions of one agent may be affected by the other agents' actions, and the agent's accumulated reward is a function of all agents' joint action.

Let  $L(s, a)$  and  $U(s, a)$  be the lower and upper bound on the Q values of a MMDP,  $L^i(s_i, a_i)$  and  $U^i(s_i, a_i)$  be the lower and upper bound on the  $i$ th MDP's Q values and  $Q_i^*(s_i, a_i)$  be the optimal value function of the  $i$ th MDP, then it is true that  $L^i(s_i, a_i)$ ,  $U^i(s_i, a_i)$  and  $Q_i^*(s_i, a_i)$  are the same. Based on the idea, we transform the optimal value functions of each subtask and propose an online learning algorithm LU-Q similar to [4], but the optimal value functions of each subtask is learned by MAXQ-Q [1]. Pseudo-code of LU-Q is shown in table 1.

LU-Q first computes the actions that have the minimal Q values in line 3 and initializes  $L(s, a)$  and  $U(s, a)$  in step 6 and 7. When a joint state is visited, the algorithm chooses a joint action  $a$  from the action set  $A(s)$  (action set in state  $s$ ) using an action selection policy derived from  $U(s, a)$ . The backup operations in step 17 and 18 are used to update the lower and upper bound on the chosen state-action value functions.  $\bar{L}$  is the maximal lower bound value, and a new action set in state  $s$  is computed in line 20 by pruning actions whose upper bounds are less than  $\bar{L}$ .

**Table 1.** Online multiagent learning algorithm LU-Q

---

1	for all $a \in A, s \in S$
2	for all $i \in M$
3	$b_i = \arg \min_{a_i \in A_i(s_i)} Q_i^*(s_i, a_i)$
4	$j = \arg \max_{i \in M} Q_i^*(s_i, a_i)$
5	end for
6	$L(s, a) = \max_{i=1}^k Q_i^*(s_i, a_i) - \min_{a_j \in A_j(s_j)} Q_j^*(s_j, a_j)$
7	$U(s, a) = \sum_{i=1}^k Q_i^*(s_i, a_i) - \sum_{i=1}^k Q_i^*(s_i, b_i)$
8	end for
9	do until algorithm converges
10	$s_0 = (s_{1,0}, s_{2,0}, \dots, s_{k,0})$ the starting states // $s_{i,0}$ is starting state of $i$ th MDP
11	$s = s_0$
12	computing action set $A(s)$
13	while $s$ is not terminal state
14	selecting an action by any given exploration policy using $U(s, a)$
15	$L' = \max_{a \in A(s)} [(1 - \alpha(t))L(s, a) + \alpha(t)(\sum_{i=1}^k r(s'_i   s_i, a_i) + \gamma \max_{a' \in A(s')} L(s', a'))]$
16	for all $a \in A(s)$ // $\alpha(t)$ is learning rate
17	$L(s, a) = (1 - \alpha(t))L(s, a) + \alpha(t)[\sum_{i=1}^k r(s'_i   s_i, a_i) + \gamma \max_{a' \in A(s')} L(s', a')]$
18	$U(s, a) = (1 - \alpha(t))L(s, a) + \alpha(t)[\sum_{i=1}^k r(s'_i   s_i, a_i) + \gamma \max_{a' \in A(s')} U(s', a')]$
19	end for
20	$A(s) = \cup a \wedge U(s, a) \geq L'; s \leftarrow s'$
	end while
	end do

---

In a single agent domain, it is clear  $L(s, a) = U(s, a) = Q^*(s, a) - \min_{b \in A(s)} Q^*(s, b)$ , and both  $L(s, a)$  and  $U(s, a)$  are no less than 0. So, the initialization of  $L(s, a)$  and  $U(s, a)$  in line 3,4,6,7 keeps the property that both  $L(s, a)$  and  $U(s, a)$  are no less than 0. Line 7 shows that  $U(s, a)$  is the sum of  $k$  positive terms, and  $L(s, a)$  is the maximal value of the  $k$  terms. We have  $U(s, a) \geq L(s, a)$ . This shows upper and lower bound are well defined.

LU-Q has no constraint on the individually learned optimal value functions of each subtask. Initialization in table 1 shows whether  $Q_i^*(s, a)$  is positive or negative, we always have  $U(s, a) \geq L(s, a)$ . So we say, the action set will not be empty in each valid state.

One obvious advantage of LU-Q is that it prunes actions whose best lower bound is larger than upper bound and backup operations are applied to those left actions. If only one action is left, regardless of whether agent's upper and lower bound is convergent, the optimal policy for that state is obtained.

Learning of the value functions is just an approach to finding the optimal policy, so if the definition of value functions is reasonable, it might be used to learn an agent's optimal policies of a task. In table 1, we define  $U(s, a)$  the upper bound on value function, and exploration policy is derived from it. This definition can be explained as follows. If there is no interaction between agents, each agent learns individually without any constraint, then the agents' joint optimal reward can be computed as  $Q^*(s, a) = \sum_{i=1}^k Q_i^*(s_i, a_i)$ . We have  $U(s, a) = Q^*(s, a) - \sum_{i=1}^k \min_{b_i \in A_i(s_i)} Q_i^*(s_i, b_i)$ . If we use  $U(s, a)$  to derive the exploration policy, we can get the same optimal policy as that gotten by exploration policy derived from  $Q^*(s, a)$ . If there is interaction between agents, constraints on the agents' actions may appear, and we have  $Q^*(s, a) \leq \sum_{i=1}^k Q_i^*(s_i, a_i)$ . We can get  $Q^*(s, a) - \sum_{i=1}^k \min_{b_i \in A_i(s_i)} Q_i^*(s_i, b_i) \leq U(s, a)$  (from line 7), so we say definition of upper bound is reasonable.

We define the value functions of a cooperative multiagent system as the accumulated rewards of the agents' joint actions. In the worst case, all agents except for  $j$  gets the lowest value functions, and  $j$  gets the largest value functions. More formally,  $\forall i (\neq j) \in M$ ,  $Q_i^*(s_i, a_i) = \min_{b_i \in A_i(s_i)} Q_i(s_i, a_i)$ ,  $Q_j^*(s_j, a_j) = \max_{j=1}^k Q_j^*(s_j, a_j)$ . Agent  $j$  selects  $a_j$  in state  $s_j$ , and we get the following inequality  $Q^*(s, a) - \sum_{i=1}^k \min_{b_i \in A_i(s_i)} Q_i^*(s_i, a_i) \geq L(s, a)$  (from line 6). So we can say definition of lower bound is reasonable too.

$\epsilon$ -greedy and Boltzmann distribution can be the candidates of action exploration policy which is derived from upper bound. LU-Q prunes actions for current state from the action set for those actions' upper bound is worse than the best lower bound. Therefore, after a state is visited and backup operation is applied to state action value functions, some actions of that state might be eliminated. Thus greedy exploration policy would be greedy with respect to both  $L(s, a)$  and  $U(s, a)$ .

P1				P2
T1				
				T2
D2				D1

T1: taxi 1  
T2: taxi 2  
P1: position of passenger 1  
P2: position of passenger 2  
D1: destination of passenger 1  
D2: destination of passenger 2

Fig. 1. Taxi problem

### 4 Experimental Results

We use the well-known taxi problem [1] with a modified version. As shown in Fig. 1, two taxies T1, T2 and two passengers P1, P2 inhabit in the 5-by-5 grid world. The taxi problem is episodic, and in each episode, each taxi starts in the starting square. The passenger in location P1 wishes to be transported to location D1, and the passenger in location P2 wishes to be transported to location D2. One episode can be described as follows. The taxi goes to the location of one passenger, picks up the passenger and goes to the location of the passenger, and then drops down the passenger at the destination location. Agent learns to find the closest passenger and the shortest path to transport it.

Black lines in Fig.1 are walls. If an agent hits the wall, it stays at its previous location, and gets a reward of  $-1$ . 6 actions are allowed for each agent(four navigation actions, pickup and dropdown passenger), and each of the four navigation actions gets a reward of  $-1$ . If an agent picks up a passenger at the correct location, it gets a reward of  $-1$ , else, it gets a reward of  $-10$ . If an agent drops down a passenger correctly, it gets a reward of  $+20$ , or else, it gets a reward of  $-10$ .

Each agent has 75 states(25 grid location, 3 taxi status: empty, carrying passenger P1, carrying passenger P2). For this domain, the number of state-action pair is  $75 \times 75 \times 6 \times 6$ . If we apply standard Q learning algorithm to this problem, convergence of Q values is very slow.

We adopt Boltzmann distribution as the action exploration policy, and set temperature and temperature decaying rate to be 190 and 0.9985. Learning discount rate is set to be 0.99. One episode is over after both P1 and P2 are transported to D1 and D2 respectively. In Fig. 2, the horizontal axis represents the number of episodes, and the vertical axis represents the average number of steps in an episode of ten trials. Both agents have already learned their optimal value functions individually by using MAXQ-Q. Results show LU-Q learns faster than standard Q learning.

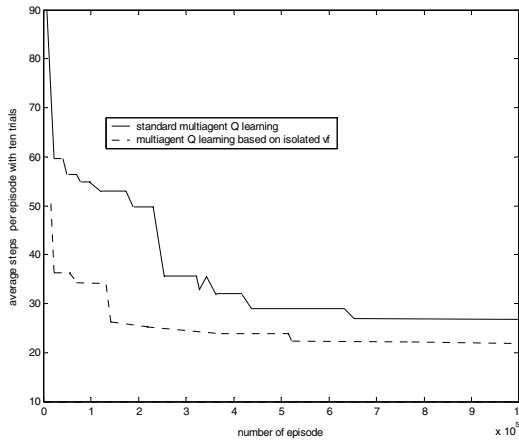


Fig. 2. Performance comparison of standard Q learning with LU-Q

Some other experiments have been done to test the performance of LU-Q. We adopt MAXQ-Q [1] to learn each subtask's optimal value functions. Before the convergence of these optimal value functions, we use these non-optimal values in LU-Q. Experimental results show even the values of each subtask is not convergent, LU-Q learns the agents' optimal policies faster than standard Q learning does too.

## 5 Conclusions and Future Work

Multiagent Q learning has aroused great interest in artificial intelligence. Algorithms proposed by Littman [6], Hu et al [7] and Greenwald [8] just focus on stochastic Markov game problems. When the state-action space becomes larger, learning becomes more intractable. We propose a planning algorithm LU-Q similar to value iteration, and learning of multiagent's upper and lower bound is executed by merging agents' optimal value functions of each subtask through MAXQ-Q. Performance comparison with standard Q learning shows LU-Q learns faster.

For a multiagent system, if all agents work together as a team, coordination is needed. To simplify learning of multiagent's optimal policies, the agents' individually learned knowledge is quite useful. In the future work, we intend to use LU-Q to solve complex multiagent problems.

## References

1. Dietterich, T.G.: Hierarchical Reinforcement Learning With the MAXQ Value Function Decomposition. *J. of Artificial Intelligence Research*. 13(2000)227-303
2. Watkins, C. J. C. H.: Learning from Delayed Rewards. Cambridge, UK: Cambridge University. Ph.D. thesis(1989).
3. Singh, S., Cohn, D.: How to Dynamically Merge Markov Decision Processes. In: 17th Int. Conference on Neural Information Processing Systems(1999)
4. Ghavamzadeh, M., Mahadevan, S.: A Multiagent Reinforcement Learning Algorithm by Dynamically Merging Markov Decision Processes. In: 1st Int. Joint Conference on Autonomous Agents and Multiagent Systems, Bologna( 2002)
5. Boutilier, C.: Sequential Optimality and Coordination in Multiagent Systems. In: 16th Int. Joint Conference on Artificial Intelligence, Stockholm(1999) 478-485
6. Littman, M.L.: Markov Games as a Framework for Multi-Agent Reinforcement Learning. In: 11th Int. Conference of Machine Learning, New Brunswick(1994) 157-163
7. Hu, J., Wellman, M.P.: Nash Q-Learning for General-Sum Stochastic Games. *J. of Machine Learning Research* 1 (2003) 1-30
8. Greenwald, A., Hall, K., Serrano, R.: Correlated-Q Learning. In: 20th Int. Conference on Neural Information Processing Systems, Workshop on Multiagent Learning(2002)

# The Compression of Massive Offline Relations\*

Jizhou Luo, Jianzhong Li, Hongzhi Wang, Yanqiu Zhang, and Kai Zhao

Computer Science Department, Harbin Institute of Technology, China  
{luojizhou, lijzh, wangzh, sarai, zhaokai}@hit.edu.cn

**Abstract.** Compression database techniques play an important role in the management of massive data in database. Based on an important feature of offline relations and the features of operations on these data, we propose a compression method of massive offline relations to improve the processing performance of these relations. Experiments show that our method is efficient.

## 1 Introduction

Massive data often appears. For examples, the data generated by high-energy-density physics experiments in Lawrence Lab can be up to 300T bytes per year. The data of a large telecommunication company's business data gets up to TeraBytes per month. In last three decades, the data accumulated in demography, geognosy, meteorology and nuclear industries is even up to 1PetaBytes.

The efficiency is the key issue while processing massive data. Many results about the storage, management and query processing of massive data have been obtained. Many people are invoked in finding efficient algorithms or techniques to deal with massive data; some other people adapt the compression techniques to the storage, management and query processing of massive data in database, and these works lead to the compression database techniques [1-9].

In general, massive relations in database can be divided into two kinds. One consists of massive online relations (*MONLRs*) that are being used frequently in manufacturing currently. So, the goal of compressing *MONLRs* is to reduce the costs of operations on these relations and guarantee the performance of the database [1-9]. The other consists of the massive offline relations (*MOFLRs*) that are not used frequently now. But for some purpose, we must keep them for a long time. Because the storage space of online database is finite, these *MOFLRs* have to be stored offline. However, the types of operation on *MOFLRs* are exiguous and the frequency of those operations is low. Additionally, the offline data will accumulate as time goes. So we must consider the following two issues while compressing *MOFLRs*: to reduce the storage requirements as possible and to ensure the performance of these limited queries. So far, we have not see any literature about the compression of *MOFLRs*. In industries, the usual method is to export the data of database into text files and com-

---

\* Supported by the National Natural Science Foundation China under Grant No.60273082.

press them with tools such as *Winzip*, *Compress* etc. The compressed files are stored on the tertiary storages. Before queries being executed, these files must be decompressed and imported back into database. So the queries on *MOFLRs* are costs intensive operations and their performance is unacceptable for most users. In this paper, we propose a compression strategy for *MOFLRs* to overcome the shortcomings.

## 2 Compression Algorithm of *MOFLR*

In contrast to *MONLRs*, the features of *MOFLRs* observed are as follows. Firstly, the sizes of *MOFLRs* are far larger and will grow rapidly. Secondly, only a few of *MOFLRs* grow rapidly as the time goes. Thirdly, the queries on *MOFLRs* are quite different, the types of queries on *MOFLRs* are much less and their frequencies are also much lower. Finally, most attributes of *MOFLRs* do not appear in query conditions.

To improve the performance of the queries, we classify the records of each *MOFLR* into many subsets. All records in each subset satisfy some common conditions. And each subset can be compressed into files with any compression algorithm and stored onto the tertiary storage. When a query is executed, we first find out the subsets involved and decompressed them. The query can be completed in two strategies. One is to import the decompressed files back into the database and to process the query as usual. The other is to filter the records in the decompressed files directly according to the conditions appearing in the query. The server of management system will choose a lower cost strategy to complete the query.

Additionally, together with the small relations or the relations with stable sizes, the information about classification is stored in the database of sever of the system. This arrangement is to save I/O costs between the secondary storage and the tertiary storage further. The architecture of the management system has following three advantages. The access of data has randomness in some extent. and the I/O costs for each query may be small. At last, the compression algorithm can be chosen flexibly.

Now, we consider the classification of records. Since most types of queries on *MOFLRs* are known before compressing. The atomic predicates appearing in *where-clause* of these queries can be extracted and so do the predicates associated with a specific relation *R*. These predicates can be divided into three kinds, i.e. the predicates used as join conditions, **indefinite predicates**, such as “user-name = ???”, which are known definitely until the queries will be executed, and **definite predicates**, such as “a phone-call last at least 5 minutes”, which known definitely according to the applications on *MOFLRs*. It’s hard to classify records according to the first two kinds of predicates. So, we use the third kind of predicates to classify the records of *R*.

Suppose the set of definite predicates associated with *R* be  $PRED_R = \{pred_1, pred_2, \dots, pred_m\}$ . They are not independent however. For example, if a record satisfies predicate “a phone-call last at least 10 minutes”, then it also satisfies predicate “a phone-call last at least 5 minutes”. If every record satisfying predicate  $pred_i$  also satisfies predicate  $pred_j$ , then we denote this relation as  $pred_i \leq pred_j$ . To reduce the



number of subsets, we first eliminate such relation from  $PRED_R$ . If  $pred_j, pred_j \in PRED_R$  and  $pred_i \leq pred_j$ , we delete predicate  $pred_i$  from  $PRED_R$ . Finally, we denote the set of left predicates with the same symbols. Usually,  $m$  is no more than 10.

For each subset, a mask with  $\lfloor \log_2 m \rfloor + 1$  bits length is needed, indicating which predicates in  $PRED_R$  the records in the subset satisfy. For example, the  $i_{th}$  bit of the mask for some subset is 1 if all records in the subset satisfy predicate  $pred_i \in \{pred_p, \dots, pred_m\}$  and 0 otherwise.

To complete the classification, we must decide in which subset each record  $r$  of  $R$  should be put into. We use the following formula to compute the mask  $M(r)$  of record  $r$  to indicate the predicates that  $r$  satisfies and  $r$  is put into the subset with mask  $M(r)$ .

$$M(r) = \sum_{i=1}^{\lfloor \log_2 m \rfloor + 1} 2^{i-1} \chi_r(pred_i), \quad \text{where } \chi_r(pred_i) = \begin{cases} 1 & \text{if record } r \text{ satisfies predicate } pred_i \\ 0 & \text{otherwise} \end{cases}$$

If we compute the mask for each record  $r$  while compressing, the speed of compression will be very low. So we let the data producer compute these masks and store them in the online database together with the data itself. These operations do not force a speed-decrease in the online database.

Now, let's consider the compression of massive relation  $R$ . Since the memory size  $M$  is much less than the size of  $R$  or the size of each subset, we have to compress each subset into many files. In order to access the data files fast in the processing of queries, we record the compressed files' addresses corresponding each subset. So we create a relation  $MAPPING \langle mask, adress \rangle$  in the database on these secondary storage. Each time when we write a compressed file onto the tertiary storage, a tuple will be inserted into  $MAPPING$ . Moreover, to reduce the seek time on the tertiary storage, the size of each file should not be too small. So, we suppose the size of each data file before compressing be  $SIZE$ .

#### Algorithm 1: compress\_M\_R

**Input:** relation  $R$ , the set of definite predicates  $\{pred_p, pred_d, \dots, pred_m\}$ , the size of available memory, the size  $SIZE$  of each data file, relation  $MAPPING$ ;

**Output:** the compressed storage of  $R$  on the tertiary storage.

1. Allocate  $2^m$  memory buffers  $B[0:2^m-1]$ , each with size  $M/m$ ;
2. FOR each record  $r$  of  $R$  DO
  - 2.1 Insert  $r$  into buffer  $B[M(r)]$ ;
  - 2.2 IF  $B[M(r)]$  is full THEN
    - 2.2.1 Write  $B[M(r)]$  onto disk;
    - 2.2.2.1 Compressing the data file  $F_{B[M(r)]}$ ;
    - 2.2.2.2 Write the compressed data file onto the tertiary storage and suppose the address of the file is  $address$ ;

We allocate a memory buffer for each subset, each with size  $M/m$ . And for each buffer, there is a file with size  $SIZE$  on the disk. While compressing  $R$ , we deal with its records one by one. According to  $M(r)$ , we can put record  $r$  into a proper buffer. If the buffer is full, we write its data into the corresponding data file. Repeat this process until the size of one data file reaches  $SIZE$ . Then compress the data file and write it onto the tertiary storage. Finally, insert the corresponding tuple into

*MAPPING*. Repeat these operations until all records of  $R$  are processed. The algorithm scans  $R$  one pass, need one I/O between disk and the tertiary storage and two I/O between memory and disk, which results that the algorithm is slower than the traditional method. However the compression is executed only once, it is devisable if the performance of query processing is improved. Additionally, since  $m$  is very small, we need not fear the lack of memory. Moreover, the compression ratio of the algorithm depends on the compression algorithm chosen in step 2.2.2.1.

### 3 Query Processing in the Compressed Data

The key issue of query processing is to find out the data files involved on the tertiary storage. We begin with the queries on one massive compressed relation  $R$  without join operations with other relations. Let  $Q$  be such a query. Suppose the set of definite predicates appearing in query  $Q$  is  $\{pred\_Q_1, pred\_Q_2, \dots, pred\_Q_s\} = PRED(Q)$ . According to the relationships among these predicates, we can construct a binary tree  $T$  whose leaf nodes are these predicates and internal nodes are *AND* or *OR*.

According to the construction of  $PRED_R$ , we know that there is a definite predicate  $pred_{j_i} \in PRED_R$  such that  $pred\_Q_i \leq pred_{j_i}$  for any  $pred\_Q_i \in PRED(Q)$ . To find out the data files we must decompress, we have to substitute  $pred_{j_i}$  for  $pred\_Q_i$  in the binary tree  $T$ . For simplicity, we still denote the modified binary tree as  $T$ . From  $T$ , we can know the subsets in which the records satisfying query  $Q$  lie.

Let the set of masks of the subsets satisfying the definite predicates in the right sub-tree of root is  $S_1$  and that corresponding the left sub-tree is  $S_2$ . If the operator in root is *OR*, then the set of masks of the subsets satisfying the definite predicates in  $T$  is  $S_1 \cup S_2$ . If the operator in root is *AND*, then the set of masks is  $\{mask_1 \wedge mask_2 | mask_1 \in S_1, mask_2 \in S_2\}$ . So the set of masks can be computed iteratively.

If the output of algorithm MASK is  $S$ , then the compressed data files whose masks are in  $S$  are those we must decompress. For  $\forall mask \in S$ , we can find out the data files' addresses with mask  $mask$  from *MAPPING* and then access the file and decompress it. The server of the management system will chose one of strategy described in section 3 to obtain the query results.

#### Algorithm 2: MASK

**Input:** the root  $root$  of the modified binary tree  $T$ ;

**Output:** the set of masks of subsets satisfying the predicates in  $T$ ;

1. IF  $T$  is a leaf in which the definite predicate is  $pred_{j_i}$ 
  - a. THEN return  $\{0 \dots 010 \dots 0\}$  the  $j_{th}$  bit is 1 and other bits are 0s.
2.  $S_1 = MASK(root \rightarrow right\ child)$
3.  $S_2 = MASK(root \rightarrow left\ child)$ ;
4. IF the operator in  $root$  is *OR* THEN return  $S_1 \cup S_2$ ;

Suppose there are  $m$  definite predicates in  $PRED_R$  and  $f$  compressed data files in each subset. Given a query  $Q$ , suppose there are  $d$  definite predicates in  $Q$ . Let  $T_o$ ,  $T_c$  denote the run time of  $Q$  in original compression architecture and the runtime of  $Q$  in our compression architecture respectively.

**Theorem.**  $T_c/T_o \leq 1/2^{m-d}$

If the definite predicates in query  $Q$  are a small part of all predicates used in compression, the performance of  $Q$  can be improved notably. However, if all definite predicates in  $PRED_R$  appear in  $Q$ , then we have to decompress all data files and the performance is the same as the traditional method.

If there are join operations between pairs of relations in query  $Q$ , we process  $Q$  in four steps. Firstly, find out all compressed massive relations in  $Q$ . Secondly, extract definite predicates associated with each compressed massive relation from  $Q$ . Thirdly, utilize the method above to obtain medial results. Finally, load these medial results into database on the secondary storage and execute query  $Q$ .

### 4 Experiments

We implement our system and give the experimental results here. All experiments run on a 2.0GHz Intel Pentium processor with 256M main memory running Linux 6.2. The experiments are conducted with real data sets of a telecommunication provider. The data set used is a 10G subset of the massive relation. According to the application, we use six definite predicates. We look into the effects of compression algorithm on compression ratio, the comparison between the costs of the traditional method and that of our method, and the comparison of the performance of these two methods.

In step 2.2.2.1 of compress\_M\_R, we use Adaptive Huffman coding, Run length coding, LZW, and Arithmetic coding to compress the data set respectively. The compression ratios are listed in fig.1. The horizontal axis denotes the value of parameter *SIZE* in the algorithm.

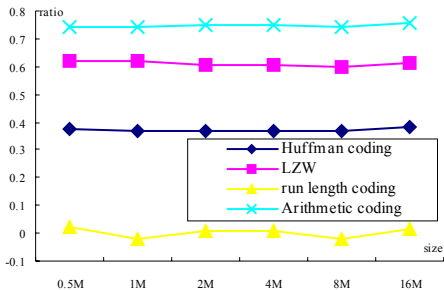


Fig. 1. Compression ratios

We allocate 2M buffer for each subset and run the compression algorithm. The costs are listed in table 1. We noted that the time of this algorithm with  $SIZE=1M$  or  $2M$  is very close to the time of the traditional method. This is because there are no additional I/O between memory and disks. On the contrary, when the additional I/O is necessary, the runtime of our algorithm is longer. In order to save the seek time of accessing data files, this value of  $SIZE$  should be as large as possible. So, we set  $SIZE=16M$  which is the maximum value of our system.

After compressing data set, we run four queries on the compressed data. The run-time these queries is listed in table 2. The third column and last column are the run-times of our method with different strategy mentioned in section 3.

**Table 1.** The costs of compression

SIZE	The costs of compression (s)	
	Old method	New method
<b>1M</b>	41275	42943
<b>2M</b>	40524	41780
<b>4M</b>	39265	51325
<b>8M</b>	37932	49756

**Table 2.** The performance of query processing

	The costs of query processing (s)		
	Old method	New import	New filter
Sql1	46353	18535	17362
Sql2	45119	17683	16408
Sql3	43675	9172	10803
Sql4	46379	20294	16895

## 5 Conclusions

We have notice that the compression of *MOFLRs* is quite different from the compression of massive online relations. On the basis of the characteristics of *MOFLRs*, we have proposed a management system and designed a compression algorithm of *MOFLRs*. The theoretical analysis and the experimental results indicate that our method can improve the performance of query processing efficiently, although the costs of compression may increase.

## References

- Jonathan Goldstein, Raghu Ramakrishnan, and Uri Shaft. "Squeezing the most out of relational database systems". In *Proc. of ICDE*, page 81, 2000.
- Wee K.NG and CHINYA V Ravishhankar. "Block-Oriented Compression Techniques for Large Statistical Databases". *IEEE Transactions on Knowledge and Data Engineering*, Vol.8, Match-April, 1997.
- M.A.Roth and S.J.Van Horn. "Database compression", *SIGMOD Record*, Vol 22, No.3, September 1993.
- T.Westmann, D.Kossmann et al. "The Implementation and performance of Compressed Database". *SIGMOD Record*, Vol 29, No.3, September 2000.
- S.Babu, M.Garofalakis and R.Rastogi. "*SPARTAN*: A Model Based Semantic Compression System for Massive Data Tables". *ACM SIGMOD*, May 2001.
- Jianzhong Li, Doron Rotem, and Jaideep Srivastava. "Aggregation algorithms for very large compressed data warehouses". In *Proc. of VLDB*, pages 651 – 662, 1999.
- G.Ray, J.R.Harisa and S.Seshadri, "Database compression: A Performance Enhancement Tool", *Proc. COMAD*, Pune, India, December 1995
- S.J.O'Connell and N.winterbottom. "Performing Joins without Decompression in Compressed Database System", *SIGMOD Record*, Vol 32, No.1, March 2003
- Meikel Poess and Dmitry Potapov. "Data compression in oracle". In *Proc. of 29<sup>th</sup> Conference of VLDB*, 2003.

# Finding Plagiarism Based on Common Semantic Sequence Model

Jun-Peng Bao, Jun-Yi Shen, Xiao-Dong Liu, Hai-Yan Liu, and Xiao-Di Zhang

Department of Computer Science and Engineering, Xi'an Jiaotong University,  
Xi'an 710049, People's Republic of China  
baojp@mail.xjtu.edu.cn

**Abstract.** It is one of key problems in Text Mining to find document features. The string matching model and global word frequency model are two common models. But the former can hardly resist rewording noise, whereas the latter cannot find document details. We present Common Semantic Sequence Model (CSSM) and apply it to Document Copy Detection. CSSM combines the ideas of 2 models above, and it makes a trade-off between a document global features and local features. CSSM calculates the common words proportion between 2 documents semantic sequences to make a plagiarism score. A semantic sequence is indeed a continual word sequence after the low-density words are omitted. With the collection of 2 documents semantic sequences, we can detect plagiarism in a fine granularity. We test CSSM with several common copy types. The result shows that CSSM is excellent for detecting non-rewording plagiarism and valid even if documents are reworded to some extent.

## 1 Introduction

We can easily fetch more and more documents from the Internet or a Digital Library. Sometimes we may find that two document are almost the same content, though their file name are different. It is called plagiarism if the copy is illegal or unauthorized. We often need to know whether some part or the whole of the given document is the copy of other documents. This work is called Document Copy Detection.

In this paper, we introduce a Common Semantic Sequence Model (CSSM) that can find plagiarism effectively even if the document is reworded to some degree. There are two basic DCD schemes, one is string matching scheme, such as [2,4,5], and the other is global word frequency scheme (i.e. bag of word models), such as [7,8]. The string matching approach can find exactly which string is copied. It is a great disadvantage of this detection scheme that changing some words of string can easily cheat it. The word frequency approach, which is derived from Vector Space Model (VSM) [6], can find partial sentence copy and has better performance against noise, but its false positive is higher than that of the above, as reported in [7]. It is because that some documents that have the same (or similar) topics often contain the same key words. It is difficult to exactly locate plagiarism positions by this method.

The fingerprints represent the local features of a document, whereas the word frequencies represent the global features. Therefore, it is better to take account of both global and local features in order to detect plagiarism in certain detail. In CSSM, we

first find out the semantic sequences based on the concept of semantic density, which reflects the locally frequent semantic features, and then we collect all of the semantic sequences to imply the global features of the document. Finally, we calculate the similarity of 2 documents by counting the common words between their features.

## 2 Common Semantic Sequence Model

### 2.1 Semantic Sequence

We present the detail definition of semantic sequence in [1]. In fact, a semantic sequence in  $S$  is a *continual* word sequence after the low-density words in  $S$  are omitted. Here *continual* means that if two words are adjacent in  $S[i]$ , then the difference between their positions in  $S$  must not be greater than a threshold ( $\varepsilon$ ).

A long  $S$  may have several semantic sequences. We denote all of the semantic sequences in a document  $S$  by  $\Omega(S)$ , which then includes the global and local semantic features as well as local structural information. If some sentences about question  $q$ , which only occur in a paragraph  $P$ , take very small proportion in the whole document, then words about  $q$  may seldom appear in the global features by means of global word frequency model. But they can be caught in the semantic sequences of  $P$ . Hence we believe the semantic sequence can detect plagiarism in a fine granularity so that we can find partial copy files well.

### 2.2 Selecting Features

There is a fact for DCD that if one section is shared between two documents, then we think they are a copy pair. That is to say we need not find all of the common sections between two documents in order to decide whether they involve plagiarism or not. It is enough for DCD to compare similarity in several most likely common sequences. We know that the larger number of common words there are between two strings, the more similar they are. Therefore, we select candidate semantic sequences in  $\Omega(S)$  and  $\Omega(T)$  according to the number of common words between them.

Let  $S[i] \cap T[j]$  be the set of common words between  $S[i]$  and  $T[j]$ . Let  $CL(S, T) = [(S[i_1], T[j_1]), \dots, (S[i_n], T[j_n])]$  be the list of semantic sequence pairs on document  $S$  and  $T$ , which is sorted by  $|S[i_k] \cap T[j_k]|, (1 \leq k \leq n)$  descendingly. We denote the first  $\eta$  semantic sequence pairs by  $CL(S, T)$ . The words in  $CL(S, T)$  are not only common words between  $S$  and  $T$  but also locally frequent words. We believe that  $CL(S, T)$  reflects the local identity between 2 documents in some detail. So the next work is to calculate the similarity of semantic sequences. Based on that, we decide whether there is plagiarism or not.

### 2.3 Common Semantic Sequence Model

A semantic sequence is very like a fingerprint. If we map a semantic sequence into a hash value and match them between 2 documents, then we will get a string matching model. However, in order to avoid the disadvantages of string matching model, we

absorb the idea of bag of word model. That is we calculate the proportion of common words between two semantic sequences. We believe that this method diminish the noise of rewording greatly, because few part of rewording words have trivial effect on the word frequency distribution.

According to the principle of bag of word model, namely, the more common words there are between documents, the more similar they are, we define the plagiarism score of document  $S$  and  $T$  as follows, which is called Common Semantic Sequence Model (CSSM). The CSSM plagiarism score of document  $S$  and  $T$  is  $P_{CSSM}(S, T)$ , i.e.

$$P_{CSSM}(S, T) = \frac{2 \times \sum_{k=1}^{\eta} |S[\mathbf{i}_k] \cap T[\mathbf{j}_k]|}{\sum_{k=1}^{\eta} (|S[\mathbf{i}_k]| + |T[\mathbf{j}_k]|)}, \quad (S[\mathbf{i}_k], T[\mathbf{j}_k]) \in CL_{\eta}(S, T) \quad (1)$$

At last, we use the discriminant below to make decision.

$$f(S, T) = \text{sign}(P_{CSSM}(S, T) + b/a), \quad a, b \in R, a > 0 \quad (2)$$

Let  $\tau = b/a$ , called *discriminating threshold*. When we vary the value of  $\tau$ , we will get different detection error.

### 3 Experimental Results

In this section, we do some experiments to test our CSSM and compare it with Shingle method [2] and Relative Frequency Model (RFM) [7], which was successfully used to track real life instances of plagiarism in several conference papers and journals [3].

We collect 1116 literature papers from proceedings of ICMLC02/03 (International Conference on Machine Learning and Cybernetics), from which we make plagiarized documents. Because the trend of positive error and that of negative error are always contrary, we use F1 metric to measure the synthetic performance of models, i.e.

$$F1_{\phi}(\tau) = \frac{2(1 - E_{\phi}^{+}(\tau)) \times (1 - E_{\phi}^{-}(\tau))}{(1 - E_{\phi}^{+}(\tau)) + (1 - E_{\phi}^{-}(\tau))} \quad (3)$$

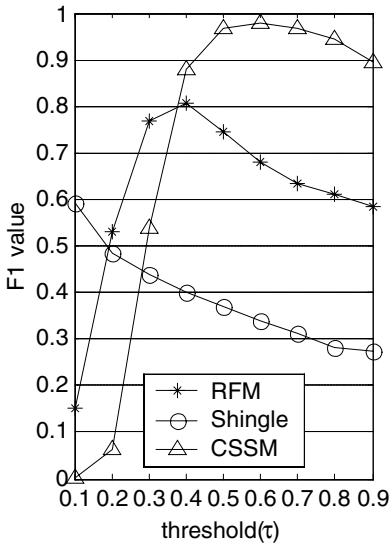
The definitions of positive and negative error as well as the detail copy types are introduced in [1].

Figure 1 shows F1 trends of RFM, Shingle and CSSM on the non-rewording corpus. Figure 2 shows the trends on the rewording corpus with each word rewording probability<sup>1</sup>  $\theta=0.7$ .

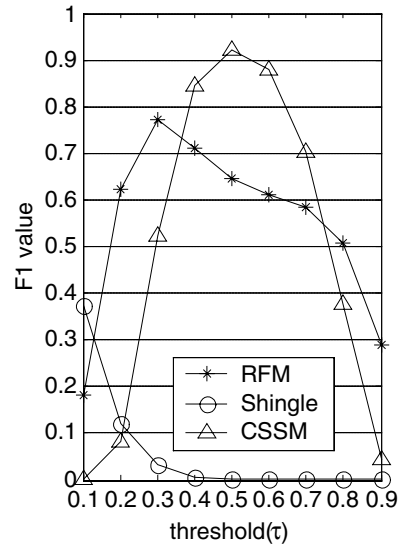
We see that the F1 values of CSSM are always greater than that of RFM and Shingle on the non-rewording corpora. CSSM is greatly better than RFM on rewording corpus. CSSM is very excellent on non-rewording corpus, whereas the Shingle model is the worst model among the 3 models no matter on which corpus. In the experiments, we find that all 3 models perform better on long texts than on short texts.

---

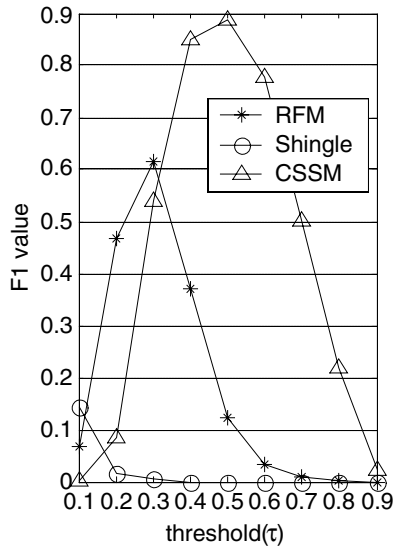
<sup>1</sup> Because the synonym set of a word contains the word itself, the real rewording probability is lower than the labeled value.



**Fig. 1.** F1 trends of RFM, Shingle and CSSM on the non-rewording corpus



**Fig. 2.** F1 trends of RFM, Shingle and CSSM on the rewording corpus with rewording probability  $\theta=0.7$



**Fig. 3.** F1 trends of RFM, Shingle and CSSM on the rewording n to 1 partial copy corpus with rewording probability  $\theta=0.7$

As we mentioned above, Shingle model [2] is a string matching approach, which randomly selects some word sequences as fingerprints from a text to construct the text's sketch, and then it compares two texts' sketches. The 2 randomly selected sketches may be some different even if from the same text. Shingle model is good at



finding very frequently repeated long word sequences, which is the main cause of its poor performance.

RFM is derived from VSM by Shivakumar and Garcia-Molina [7] in SCAM prototype. RFM exploits the similar frequency distribution words and asymmetric metric to get the containment of two texts. It is fit to find out subset copy, but it is not so valid for  $n$  to 1 partial copy. Figure 3 shows the F1 trends of 3 models on the rewording  $n$  to 1 partial copy corpus with the rewording probability  $\theta=0.7$ .

## 4 Discussions

While the discriminating threshold ( $\tau$ ) is increasing, the positive error trend is declining and negative error trend is increasing. The higher value of  $\tau$  means that we need more common words to confirm plagiarism, so that we may decrease mistakes in plagiarism detection. Consequently, when the value of  $\tau$  is low, we will get low negative error and high positive error, and when the value of  $\tau$  is high, we will get high negative error and low positive error. Altogether, the low  $\tau$  makes system radical, which misses out fewer plagiarism cases but easily misjudges the similar documents as plagiarism. In contrast, the high  $\tau$  makes system conservative, which catches serious plagiarism but may fail to detect many light or partial copy documents. Whatsoever, we can get the appropriate values of  $\tau$  (about 0.5-0.6 for CSSM) to keep both the positive and negative error in a tolerable range, where is around the peak of F1 trend.

In the short text, there is little difference among different word's frequency so that we can get few features from the short text. As the text length grows, some words' frequency is larger than others so that more features can be found out and texts are easy to be distinguished. We believe that is the reason why 3 models perform better on long texts than on short texts. Moreover, the randomly rewording action tends to decrease the frequency of high frequency words, and decrease the frequency difference among different words. Consequently, all models perform worse on rewording corpus than on non-rewording corpus. CSSM is sensitive to word local frequency. The rewording action may not only reduce the common words but also disorder the words' sequence so that CSSM may miss some plagiarized documents. That raises the negative errors of CSSM, and leads to its worst performance on the short rewording corpus.

## 5 Conclusions

The string matching model and global word frequency model both have some drawbacks on DCD. In order to get the optimal result, we have to make a trade-off between the global features and the local features. A semantic sequence is indeed a continual word sequence after the low-density words are omitted. The collection of semantic sequences in a document then contains both global and local features of the document. Moreover, we use common word counting method to diminish rewording noise. So CSSM is a hybrid model, which combines the features of word frequency model and string matching model. CSSM can detect plagiarism in a fine granularity, and experimental results show that it is superior to traditional DCD models, such as Shingle and RFM, especially on non-rewording corpus and long texts. However, the

rewording action may change the possible common words between documents, which impair the CSSM performance greatly. Whatsoever, CSSM is valid for DCD even if the document is reworded to some degree.

## Acknowledgements

Our study is sponsored by NSFC (National Natural Science Foundation of China, ID: 60173058) and Xi'an Jiaotong University Science Research Fund (ID: 573031).

## References

1. J.P. Bao, et al. Document copy detection based on kernel method. In *Proceedings of the International Conference on Natural Language Processing and Knowledge Engineering*, pp: 250-256, Oct. 26-29, 2003
2. A.Z. Broder, S.C. Glassman, M.S. Manasse, Syntactic Clustering of the Web. *Sixth International Web Conference*, Santa Clara, California USA, April 7-11, 1997.
3. P. J. Denning, Editorial: Plagiarism in the web. *Communications of the ACM*, 38(12), 1995
4. N. Heintze, Scalable Document Fingerprinting. In *Proceedings of the Second USENIX Workshop on Electronic Commerce*, Oakland, California, 18-21 November, 1996.
5. K. Monostori, A. Zaslavsky, H. Schmidt. MatchDetectReveal: Finding Overlapping and Similar Digital Documents. In *Proceedings of Information Resources Management Association International Conference (IRMA2000)*, Anchorage, Alaska, USA, 21-24 May, 2000
6. G. Salton, A. Wong, and C. Yang. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613-620, 1975
7. N. Shivakumar and H. Garcia-Molina. SCAM: A copy detection mechanism for digital documents. In *Proceedings of 2nd International Conference in Theory and Practice of Digital Libraries (DL'95)*, Austin, Texas, June 1995.
8. A. Si, H.V. Leong, R. W. H. Lau. CHECK: A Document Plagiarism Detection System. In *Proceedings of ACM Symposium for Applied Computing*, pp: 70-77, Feb. 1997.

# Web Information Extraction Based on Similar Patterns

Na Ye, Xuejun Wu, Jingbo Zhu, Wenliang Chen, and Tianshun Yao

Natural Language Processing Lab  
Northeastern University, Shenyang, China  
yn\_yena@hotmail.com

**Abstract.** Information Extraction is an important research topic in data mining. In this paper we introduce a web information extraction approach based on similar patterns, in which the construction of pattern library is a knowledge acquisition bottleneck. We use a method based on similarity computation to automatically acquire patterns from large-scale corpus. According to the given seed patterns, relevant patterns can be learned from unlabeled training web pages. The generated patterns can be put to use after little manual correction. Compared to other algorithms, our approach requires much less human intervention and avoids the necessity of hand-tagging training corpus. Experimental results show that the acquired patterns achieve IE precision of 79.45% and recall of 66.51% in open test.

## 1 Introduction

With the continuous development of the Internet, on-line resources become more and more rich, which makes the Internet a huge, hidden information library. It has become a hot research topic to accurately mine information from the plentiful semi-structured or non-structured webpages, and describe it in a regular structure so that it can be stored in relational database and made full use of. The technique of information extraction (IE) that we study in this paper is to solve the problem.

The so-called IE refers to the task of extracting useful information that users are interested in from natural language text and describing it in a structured format. Presently most IE systems perform extraction on the basis of patterns (rules). The IE process in this paper, which is a typical IE process, involves the following steps: Given an unlabeled webpage to be extracted, first remove its html labels to get plain text. Then break it into sentences and perform word segmentation, POS tagging and named entity recognition. Next filter out sentences without entities of interest. According to the patterns in the given pattern library, run pattern-matching program to verify relations among entities and finally acquire structured data and store them into database.

The construction of a pattern dictionary plays a crucial role and is a bottleneck of knowledge acquisition. Originally people construct patterns by hand. However hand-coding is too arduous. Many researchers made effort to seek an automated solution. WHISK [1], LIEP [2], and Chikashi Nobata's system [3] are among the most famous systems. In the recent years some IE methods based on bootstrapping emerged with-

out requiring hand-tagging training corpus, such as ExDisco [4] and Ellen Riloff [5]. These systems' purpose of reducing manual work is also the purpose of this paper.

## 2 Automatic Pattern Acquisition Method Based on Similarity Computation (PASC)

Here we give some related definitions before turning to describe the algorithm:

*Feature word*: Feature words are those words that appear most frequently in domain instances and can best represent domain implication. For our problem we divide feature words into two classes: first-level feature words and second-level feature words. First-level feature words must appear in the similar patterns and are assigned greater weights in the computation model; in contrast, second-level feature words do not necessarily appear and are assigned lower weights.

*Pattern*: In this paper, patterns are abstracted from named entities, feature words and the orders of them in sentences.

*Pattern Similarity*: Two patterns can be viewed as similar if they have common entity types and feature words (or synonyms of feature words), besides, their similarity computed by the following model must be greater than a given threshold.

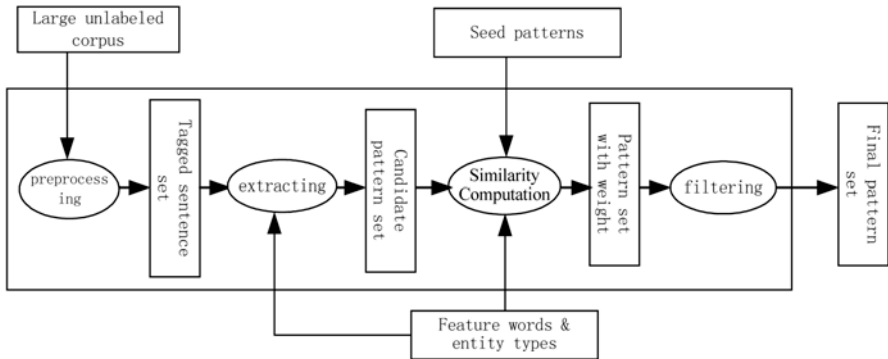


Fig. 1. Overview of the algorithm

### 2.1 Flowchart

In our experiments we preprocess the unlabeled training examples by removing html labels from training web pages, breaking text into sentences, performing word segmentation, POS tagging and named entity recognition on each sentence. Then sentences without entities of interest or feature words (or their synonyms) are thrown away. Labeled sentence set is reached.

Then system automatically abstracts labeled sentences into candidate patterns. On the basis of this the similarity between the seed pattern and candidate patterns is computed using our computation model. Similar patterns are thus obtained by discarding those with similarity below the threshold. Next the similar patterns are evaluated and filtered to reach the final set. The detailed flowchart is illustrated in figure 1.

## 2.2 Expression of Patterns

In the light of the previously mentioned definition of pattern, PASC uses a regular grammar to represent patterns.

$P \rightarrow E @ R$

$E \rightarrow \{\text{entity type}\}E \mid \{*\}E \mid [\text{first-level feature word}]E \mid <\text{second-level feature word}>E \mid \text{NULL}$

$R \rightarrow \text{num1, num2} > \text{relation}; R \mid \text{NULL}$

Each field bracketed by a pair of  $\{ \}$ ,  $[ ]$  or  $< >$  indicates a node in the pattern;  $\{*\}$  denotes wildcard whose length is limited. The content following the character “@” is the output of the pattern, that is, the relation between entities with serial number of “num1” and “num2” is “relation”.

For example, consider the sentence “微星公司生产的 K8T Neo-FIS2R 主板已经悄然在日本秋叶原上市，售价 19500 日元(The K8T Neo-FIS2R mainboard produced by MSI has quietly come into market at the price of 19500 yen in Akihabara, Japan.)”. System grows a pattern “ $\{\text{company}\} [\text{生产}(\text{produce})] \{*\} \{\text{trade}\} \{*\} <\text{上市}(\text{come into market})> \{*\} [\text{售价}(\text{price})] \{\text{money}\} \{*\} @ 1,2>\text{produce}; 2,3>\text{price}$ ”.

## 2.3 Relevant Resources

Two resources must be constructed before acquiring patterns: feature word list (and their synonyms) and seed pattern library.

Because domain feature words have powerful indicating and distinguishing ability, they are an important component in the expression of patterns. We also made use of feature word synonym resource. Therefore a list of feature words and their synonyms must be constructed.

A seed pattern library needs to be manually constructed as the input.

From the similarity computation model mentioned below, we can easily come to the conclusion that if the seed pattern library is well constructed, any accurate new pattern can find its similar pattern in the library, thus being filtered out.

## 2.4 Acquisition of Candidate Pattern

A set of patterns are abstracted from the labeled sentences according to the following basic principles:

- 1) Abstract the entities into nodes of entity type  $\{\text{entity type}\}$ ;
- 2) Feature words are retained and labeled as  $[\text{first-level feature word}]$  or  $<\text{second-level feature word}>$  nodes;
- 3) Other contiguous words are merged into wildcard nodes  $\{*\}$ ;

If the frequency of a pattern in the set is above the threshold  $\forall$ , then we add the pattern into candidate set.

## 2.5 Computing Pattern Similarity

In this section we describe the model to compute the similarity between the seed pattern and the candidate pattern. The detailed computation model is as follows:

For the seed pattern “a” and candidate pattern “s”,  $\text{Len1stKW}(a)$  and  $\text{Len2ndKW}(a)$  refer to the number of first-level and second-level feature words in “a”, respectively;  $\text{LenNE}(a)$  and  $\text{LenNE}(s)$  refer to the number of entity types in seed pattern and in candidate pattern, respectively;  $\text{Same1stKW}(a,s)$  and  $\text{Same2ndKW}(a,s)$  refer to the number of common first-level and second-level feature words (or their synonyms) between “a” and “s”, respectively;  $\text{SameNE}(a,s)$  refer to the number of common entity types between “a” and “s”.

The feature word similarity between “a” and “s” can be computed with the following formula:

$$\text{KWSim}(a,s) = \begin{cases} 0 & \text{if } \text{Same1stKW}(a,s) = 0 \\ \frac{\text{Same1stKW}(a,s) + \alpha \cdot \text{Same2ndKW}(a,s)}{\text{Len1stKW}(a) + \alpha \cdot \text{Len2ndKW}(s)} & \text{otherwise} \end{cases}$$

where  $0 < \alpha < 1$ , indicating the weight of second-level feature word.

The named entity similarity between “a” and “s” can be computed with the following formula:

$$\text{NESim}(a,s) = 2 \cdot \frac{\text{SameNE}(a,s)}{\text{LenNE}(a) + \text{LenNE}(s)}$$

The pattern similarity between “a” and “s” can be computed with the following formula:

$$\text{Sim}(a,s) = \lambda_1 \cdot \text{KWSim}(a,s) + \lambda_2 \cdot \text{NESim}(a,s)$$

where  $\lambda_1$  and  $\lambda_2$  are constants. Since pattern “a” and “s” can only be similar if they have entity types and feature words (or their synonyms) in common,  $\lambda_1$  and  $\lambda_2$  must meet the following criteria:

$$\begin{cases} \lambda_1 + \lambda_2 = 1; & \text{if } \text{KWSim}(a,s) > 0 \text{ and } \text{NESim}(a,s) > 0 \\ \lambda_1 = 0 \text{ and } \lambda_2 = 0; & \text{otherwise} \end{cases}$$

We filter the patterns assigned with similarity weights to reach the final pattern set. Patterns with similarity above threshold are filtered out. After comparing with seed pattern to decide entity relation, they are added to the final set.

## 3 Experiments and Evaluation

The experiments reported here use the pattern mentioned in section 2.2 as the seed pattern. Since there is no Chinese-like MUC corpus exists, we retrieved 559 web pages from Internet through feature words (and their synonyms) of the seed pattern. We use 500 web pages for training, and the rest 59 for testing. We use the toolkit CipSegSDK [6] for text segmentation.

Due to the limited capability of the named entity recognizer we use, part of the entities are labeled by hand. In experiments the pattern frequency threshold is set to 2.

3.1 Experiments

We let the similarity threshold  $\beta$  equal to 0.3 and made experiments to acquire new patterns with  $\lambda_1$  ranging from 0.2 to 0.9. Results are shown in table 1:

**Table 1.** Experiment results with  $\beta=0.3$  as  $\lambda_1$  varies

$\lambda_1$	0.3	0.4	0.5	0.6	0.7	0.8	0.9
acquired	22	22	22	22	28	30	30
accurate	20	21	21	21	26	28	28

We also experimented over different threshold  $\beta$  with  $\lambda_1$  set to 0.8. Experimental results are shown in table 2:

**Table 2.** Experiment results with  $\lambda_1=0.8$  as  $\beta$  varies

$\beta$	0.1	0.2	0.3	0.32	0.4	0.5	0.6
acquired	30	30	30	27	17	10	6
accurate	28	28	28	26	16	10	6

On the basis of the results in table 1 and table 2, taking both precision and recall into consideration, we set  $\lambda_1$  to 0.8 and  $\beta$  to 0.3 and acquired 28 patterns.

We also labeled patterns from the 500 training web pages by hand and acquired 31 patterns.

Table 3 shows the results of closed test over the web pages in training set and open test over the web pages in testing set with the acquired 28 patterns and the hand-crafted 31 patterns.

**Table 3.** Results of Experiment

Experiment		Automatic Pattern	Manual Pattern
Closed test	Precision	84.56%	90.33%
	Recall	89.34%	96.75%
	F-measure	86.88%	93.43%
Open test	Precision	79.45%	91.33%
	Recall	66.51%	69.36%
	F-measure	73.41%	78.84%

Where

$$F - measure = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall}$$

The experimental results indicate that IE precision and recall of automatically acquired patterns are close to that of hand-built patterns while avoiding the huge human effort.

3.2 Discussion

The pattern recognition errors are mainly caused by the following reason:

If there is more than one price entity in the example, the system cannot decide which one should be chosen as slot of pattern, for example:

{\*} {money=1} {\*} {trade=1} {\*} #售价(price)# {\*} @ 2,1>price

记者昨日获悉, 江南奥拓在我市的最低售价已跌破 3 万元, 最低售价仅为 29800 元。(The reporter leaned yesterday that the lowest price of Jiangnan Aotuo in our city has gone below 30,000 yuan, the lowest price is only 29,800 yuan.)

The above errors can be easily corrected by manual inspection.

We can imagine that with the scale of corpus increases, it will become more and more difficult for handwork to reach high recall, in contrast, automatically acquired patterns will achieve increasing amount and coverage, thus making automatic IE performance goes beyond that of manual IE.

## 4 Conclusion

Compared to traditional approaches, our similarity-based pattern acquisition method avoids the need of labeling training corpus, and takes full advantage of the feature words in patterns and their synonym resource. Throughout the pattern acquisition process, little human intervention is required and a lot of human effort is saved.

The weak point of our method lies in the lack of restriction over the order and relative position between slots and feature words, and it depressed the precision of patterns to some extent. In the future work, we will consider introducing some heuristic rules to aid and restrict the generation of patterns and improve pattern precision.

## Acknowledgments

Supported by the Key Project of Chinese Ministry of Education.(NO 104065) funded by the National Natural Science Foundation of China and Microsoft Asia Research under Grant No.60203019

## References

1. Stephen Soderland. Learning Information Extraction Rules from Semi-Structured and free text. Machine Learning, vol. 34, pages 233-272,1999
2. Scott B. Huffman. Learning Information Extraction Patterns from Examples. In Proceeding of the 1995 IJCAI Workshop on New Approaches to Learning for Natural Language Processing
3. Chikashi Nobata, Satoshi Sekine. Automatic Acquisition of Pattern for Information Extraction. In Proceeding of the ICCPOL1999
4. Roman Yangarber, Ralph Grishman, Pasi Tapanainen. Automatic Acquisition of Domain Knowledge for Information Extraction. In Proceeding of the COLING 2000
5. Ellen Riloff, Rosie Jones. Learning Dictionaries for Information Extraction by Multi-Level Bootstrapping. Proceedings of the Sixteenth National Conference on Artificial Intelligence
6. T.S. Yao, et al. 2002. Natural Language Processing- A research of making computers understand human languages, Tsinghua University Press, 2002. (In Chinese)



# TS-Cache: A Novel Caching Strategy to Manage Data in MANET Database\*

Shengfei Shi, Jianzhong Li, Chaokun Wang, and Jinbao Li

Department of Computer Science and Engineering  
Harbin Institute of Technology  
Harbin, Heilongjiang China 150001  
shengfei@hit.edu.cn

**Abstract.** With the development of computer networks and wireless communication technologies, more interests have been addressed on *mobile wireless ad hoc networks* (MANET) that are constructed by only wireless hosts. Previously proposed data management researches in traditional mobile database put emphasis only on the mobility and power limitation of mobile clients. In MANET database system, research must consider issues both on clients and servers. Wireless sensor networks can be seen as data source in many applications based on MANET. In these applications, mobile clients query the data generated by wireless sensor networks through mobile servers. Up to now, few papers have discussed the methods of data management in such complicated system which includes mobile clients, mobile servers and wireless sensor networks. In this paper, a novel and complicated system architecture is proposed. In order to deal with the data management problem, Ts-Cache strategy which is based on time-series analysis theory is also proposed, which can significantly reduce the workload of communication and improve the performance of the system.

## 1 Introduction

With the development of computer networks and wireless communication technologies, more interests have been addressed on *mobile wireless ad hoc networks* (MANET) that are constructed by only wireless hosts. In contrast with the traditional mobile network, in MANET, every host plays the role of a router and communicates with each other without the support of base station. All nodes are wireless and battery powered [1].

Previously proposed data management researches in traditional mobile database put emphasis only on the mobility and power limitation of mobile clients [1-10]. In MANET database system, research must consider issues both on clients and servers, because they are all mobile and power limited.

In this paper, a novel and complicated system architecture is proposed. In order to deal with the data management problem, Ts-Cache strategy which is based on time-

---

\* This work was supported by NSF of Heilongjiang Province of China under the grant No. ZJG03-05.

series analysis theory is also put forward, which can significantly reduce the workload of communication and improve the performance of the system.

The contribution of this paper lies in two aspects: the architecture to represent MANET based Mobile Distributed Database and some formal descriptions of the probability model of data and time-series based cache model; the investigations to find ways in which a time-series cache can be used efficiently in many MANET and wireless sensor networks applications.

## 2 Model of Hybrid MANET-Based DMDB System

### 2.1 Formal Definition of HM-DMDB

**Definition 1.** The HM-DMDB is defined as a HM-DMDB::=(WS, MS, MC, H-Ad). Where WS is wireless sensor node, MS represents mobile servers, MC stands for mobile client, and H-Ad is hybrid wireless ad-hoc network which includes both mobile Ad-Hoc network and static wireless sensor networks.

**Definition 2.** The probability model of data in HM-DMDB is defined as PModel ::= (Attr, Location, TS-M, CR, Tstamp). Where Attr is the attribute of sensor, Location identifies the location of data sensed by a cluster of sensors, TS-M is time-series model of data such as AR or ARMA, etc, for the simplicity, we use AR model to represent the time series data generated by sensor networks; CR is confidence region of error bound between actual value of data and predicted value from TS-M with specific probability, and Tstamp is time stamp in which the TS-M is established.

AR(n) model can be represented as follow:

$$X_t = \varphi_1 X_{t-1} + \varphi_2 X_{t-2} + \dots + \varphi_n X_{t-n} + a_n \quad (1)$$

where  $a_n$  is a White-Noise series, and  $a_n \sim NID(0, \delta_t^2)$ , without the loss of generality, we can assume the mean of the time series  $X$  is 0. Using AR model, the MC can predict the future value of time series at time  $t$  with specific probability confidence. For example, suppose that time series  $X$  satisfies AR(1) model, at time  $t$ , we can predict the  $X_{t+k}$  with 95% probability limit:

$$X_t(k) \pm 1.96 \delta_t^2 (1 + \varphi_1^2 + \varphi_1^4 + \dots + \varphi_1^{2(k-1)})^{1/2} \quad (2)$$

$X_t(k)$  can be calculated by formula(1) in which  $n=1$ .

**Definition 3.** The hierarchical TS-cache model HTS-Cache is defined as HTS-Cache::= ( Layer, Query, PModel). Where Layer is the position in the hierarchical architecture mentioned in section 2.1, Query is the query which gets results from sensor networks. PModel has been defined above and is used to answer the Query. The cache is divided into two sections: one section is the data which are the results of previous queries, the other is the data calculated from the PModel. The MC can get results locally not only from the previous real data but also from the future data to answer the new query.

**Definition 4.** Time-Series based Cache Invalid Report Model (TS-IR) is defined as  $TS-IR ::= (Attr, Location, TS-M|RawData, \Delta error, Host| Broadcast, Tstamp)$ . TS-IR is used to inform the MS or MC to change the parameters of PModel when the error of predicted value exceeds the error bound predefined. Attr, Location and TS-M are the same as defined in PModel. RawData is raw data generated by sensor. MS sends changed parameters and initial values of TS-M to MCs.  $\Delta error$  shows the difference between the data values which are calculated by old parameters and new parameters separately. When TS-IR is sent by sensor node, the destination is the host which has queried the data from this sensor node, otherwise, the MS broadcasts this TS-IR to all clients who have cached this model locally.

**Definition 5.** Special Query Based on Time-Series in HM-DMDB TS-QUERY has the following format:

SELECT (attributes| TS-Func (attributes)) FROM (Location) WHERE (predicate,  $\delta$ )  
DURATION (time interval).

Different to previous queries of wireless sensor networks, the TS-QUERY introduces time-series related functions in queries. These functions return the parameters of PModel and many other values which can only be calculated by time-series analysis, e.g., trend of the data, future values of data, etc.  $\delta$  is the error tolerance which the client can accept. For example, the client can submit a query to know the probability of temperature reaches the scope between temp1 and temp2 until the next 30 time intervals and the error bound between  $[-0.5, 0.5]$ :

select Prob(Trend(Atemperature) IN [Temp1,Temp2]) from Location2 where  $-0.5 < \delta < 0.5$  DURATION(Tcurrent+30\*Tinterval)

### 3 Data Management Strategies in HM-DMDB Using TS-Cache

#### 3.1 The Cooperation Algorithm of Construction of Probability-Based Data Model

The sensor node is responsible to answer the queries and return the results to MCs through MSs. Different to the previous methods in sensor networks, the sensor node identifies the probability model of the data with the help of MS.

At the beginning of the algorithm, because it doesn't know which model the data will be, the sensor node sends the raw data to MS using the following criteria: let  $Ex$  be the mean value of the data generated by sensor node,  $\delta$  be error tolerance defined in the query,  $x_t$  be the data at time  $t$ . If  $|x_t - Ex| > \delta$  then send  $x_t$ , otherwise the node drops it. After having received adequate data, MS can identify the model of the data and send the parameters of the model to the sensor node. For example, for the AR( $n$ ) model, the parameters include  $(n, \phi_1 \dots \phi_n, \delta)$  only. In general case,  $n$  is no more than 5. Using these parameters, the sensor node can construct the data model and use this model to filter the data more precisely. The algorithm is described in detail as follows.

**Algorithm 1: Cooperation on Construction of Dada Model****Sensor-Side:**

Register the query  $Q_i$  received from  $MS_i$ ;

When generating a data  $x_t$ , decide whether to send it to MS as follows:

If  $|x_t - E_x| > \delta$  then send  $x_t$ , else drop it;

When receiving parameters of model which is suitable for query  $Q_i$ , change data filter criteria to:

If  $|x_t - x_t'| > \delta$  then send  $x_t$ , else drop it; ( $x_t$  is the true value generated at time  $t$  and  $x_t'$  is predicted value by model);

**MS-Side:**

Send query  $Q_i$  to corresponding sensor node;

Let  $x_i$  ( $i=1$  to  $t$ ) be the data series received from sensor node at time  $t$ . Suppose the interval used to establish model is  $d$ ;

For ( $k=1$  to  $t$ )

If timestamp ( $x_{k+1}$ ) – timestamp ( $x_k$ ) is greater than  $d$ , then insert  $(x_{k+1} + x_k)/2$  between  $x_k$  and  $x_{k+1}$ ;

When receiving enough data to calculate the parameters of model, Identifies the model and send these parameters to sensor node.

Continue to receive the data from sensor for  $Q_i$  and recalculate the parameters when too much data have been accumulated;

**3.2 TS-Cache Coherence Management Algorithm**

Different to other similar works, this method puts emphasis on the following aspects:

1. The content in broadcast is mainly parameters which are in small size compared with raw data;
2. Take full advantage of TS-Cache on the router node in ad-hoc networks to reduce communication cost. Because every host node in a wireless ad-hoc network plays the role of a router, it caches some model information locally.
3. Support the mobility of MS which are never mentioned in previous methods;

When the parameter of a model has been changed, the MS will construct TS-IR to notify the other MS to change the cached model. The MS will also choose some neighbor MS to receive the TS-IR using the following selection criterias:

1. Number of received queries which are related to the changed model from the host;
2. Frequency of disconnection;
3. How much energy left

**3.3 Method to Deal with Hand-Off of MS**

Different to the servers of traditional mobile database, MS in HM-DMDB often moves from one area to another freely, which is called hand-off problem. Once a MS

roams out of the area where it currently stays, it must select a MS in this area as its agent to receive the data of sensor network corresponding to this area and to answer the queries submitted by MCs.

Every MS maintains some information about its neighbors:

1. Number of queries sent from them separately and denoted by  $N_i$ ;
2. Power remained in  $MS_i$  and denoted by  $P_i$ ;
3. Frequency of disconnection of  $MS_i$  denoted by  $F_i$ ;

Suppose  $MS_n$  will move out of the area. Before its leaving,  $MS_n$  selects another MS which satisfies the condition that the total of  $N_i + P_i - F_i$  is the maximum. After choosing a  $MS_k$ ,  $MS_n$  sends  $MS_k$  all the parameters of its model.

When  $MS_n$  moves to another area, it notifies new neighbors about its model and its frequency of mobility.

### 3.4 Query Processing Using TS-Cache

In order to save energy efficiently, in-network processing must be taken into account all the time. We can imagine that after queries have run for a longtime, TS-Caches are distributed on every node in HM-DMDB. Some of these TS-Caches are results of queries; others are relayed message on mediate node. As query in MANET must be relayed from one hop to another, it is not necessary to relay the query further when the query can be calculated and the result satisfied the precision in a mediate node. This method runs on each layer in HM-DMDB and is described in detail in Algorithm 2:

#### Algorithm 2: Query Processing Using TS-Cache

##### MC-Layer:

Whenever it submits a new query, MC first checks the TS-Cache locally;

Then the query can be divided into two parts  $Q_1$  and  $Q_2$ .  $Q_1$  is satisfied locally, whose precision of query can be satisfied by the local TS-Cache. On the contrary, TS\_FUNC of  $Q_2$  includes data in particular time in the future which can't be obtained from the local TS-Cache;

MC need to send  $Q_2$  to MS only;

Whenever MC receives a query to be routed, it takes action as above and return the results of  $Q_1$  to sender and sends  $Q_2$  to next hop;

##### MS-Layer:

Whenever a MS receives a query, no matter whether it is the destination of the query or not, it checks local TS-Cache as MC does above. The main difference between MC and MS is that MS sends  $Q_2$  to sensor networks to get the results;

##### Sensor-Layer:

Because of the limitation of capacity of sensor node, it is difficult for it to store many models locally. But the sensors can form a cluster in which related models are distributed stored. The sensor can treat the cluster as a local cache and takes actions as MC does.

## 4 Conclusion

In this paper, a novel and complicated system architecture is proposed. In order to deal with the data management problem, Ts-Cache strategy which is based on time-series analysis theory is also proposed, which can significantly reduce the workload of communication and improve the performance of the system.

Simulation results show that our algorithms could save considerable bandwidth compared with other algorithms.

In future research, we will extend our algorithm by using TS-Cache to deal with the problem of query optimize.

## References

1. Gruenwald, L, etc. Energy-Efficient Data Broadcasting in Mobile Ad-Hoc Networks. In Proc. International Database Engineering and Applications Symposium (IDEAS'02), July 2002.
2. Takahiro Hara. Replica Allocation in Ad Hoc Networks with Periodic Data Update. Third International Conference on Mobile Data Management. January 2002.
3. Imielinski, etc. Power Efficient Filtering of Data on Air. Proc. EDBT Conf., March 1994
4. Philippe Bonnet, etc. Towards Sensor Database Systems. Proceedings of the Second International Conference on Mobile Data Management January 2001.
5. Leslie D. Fife, Le Gruenwald. Research Issues for Data-Communication in Mobile Ad-Hoc Network Database System. SIGMOD Record, June 2003
6. Aksoy, D. and Franklin, M. Scheduling for Large-Scale On-Demand Data Broadcasting. In Proc. 12th International Conf. On Information Networking. January 1998.
7. Xuan, P., Sen, S., Gonzalez, O., Fernandez, J., and Ramamritham K., Broadcast on Demand: Efficient and Timely Dissemination of Data in Mobile Environments, 3rd IEEE Real-Time Technology Application Symposium, 1997.
8. Wieselthier, J., Nguyen, G., and Ephremides, A., Algorithms for Energy-Efficient Multicasting in Static Ad Hoc Wireless Networks. *Mobile Networks and Applications*, 6(4), 2001.
9. Qun Ren, Margaret H. Dunham. Using semantic caching to manage location dependent data in mobile computing. MOBICOM (2000) Baldonado, M., Chang, C.-C.K., Gravano, L., Paepcke, A.: The Stanford Digital Library Metadata Architecture. *Int. J. Digit. Libr.* 1 (1997) 108–121
10. Q Ren, MH.Dunham Using Clustering for Effective Management of a Semantic Cache in Mobile Computing. *MobiDe* 1999

# Extended Chain-Conflicting Serializability for the Correct Schedule of Transactions in Hierarchical Multidatabase\*

Guoning Chen<sup>1</sup> and Taoshen Li<sup>1,2</sup>

<sup>1</sup> Guangxi University, College of Computer, Electronic and Information,  
530004 Nanning, China  
guon\_chen@hotmail.com

<sup>2</sup> Central South University, College of Information science and Engineering,  
410083 Changsha, China  
tshli@gxu.edu.cn

**Abstract.** The issue of the correctness criterion for the schedule of transactions in hierarchical MDBS is studied in this paper. The definition and the architecture of hierarchical MDBS are firstly presented. The structure of transactions executed in this hierarchical MDBS is also presented. Then, a correctness criterion for the execution of transactions in hierarchical MDBS is proposed. An example of the application of the criterion is given as well. At last, the analysis and future development of the criterion are given, which shows the latent application of the criterion.

## 1 Introduction

A multidatabase (MDBS) is a logically integrated collection of pre-existing databases such that users can execute transactions that need to access multiple databases. It provides a feasible approach to integrate the legacy heterogeneous databases for those larger organizations that have many branches using various databases to store their specific data. One of the most concerned problems in MDBS is the concurrency control. We believe that the main job of the concurrency control in MDBS is to ensure the correct schedule of the transactions on the top global level. There are some researches about the criterion for transactions executed in MDBS have been presented, such as chain-conflict serializability [2], shared-conflict serializability [2], QSR(Quasi-serializability) [3], 2LSR(two level serializability) [4][5]. But most of these researches were carried out on the monolithic multidatabase [1]. The monolithic multidatabase is those traditional MDBSs that integrate all local databases to build a whole logical database through a single centralized software module called ‘integrity center’.

Given the rapid increase of the amount of the member databases and the distance between member databases, people find that the single monolithic MDBS architecture may lower the performance of the whole integrity system. So people choose to con-

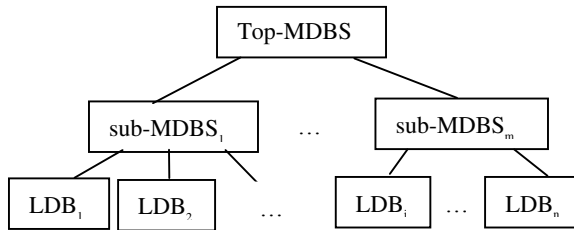
---

\* This research is support in part by the Science Foundation of Guangxi University (No. X022089) and by the Natural Science Foundation of Guangxi ( No. 9712008) .

struct the MDBS hierarchically [1]. In this paper, we dip into the concurrency control in hierarchical MDBS transactions management and present an extended chain-conflicting serializability for correct schedule of global transactions. The criterion is derived from the mostly used correctness criterion, conflict serializability, in traditional database. We prove the sufficiency of the application of the criterion in the maintenance of the serializable schedule of top global transactions. And then, an example of the usage of the criterion is also explained. Finally, we address some discussion on the work we did and predict some possible future development and application of this research.

## 2 Hierarchical MDBS Architecture and Transactions in It

Hierarchical MDBS is a MDBS that integrates local databases hierarchically according to the characteristics of member databases and the requirement of the application. But the selection of hierarchical MDBS architecture is not arbitrary, for it will have great impact on the concurrency control for transaction process [1]. For simplification, our discussions are based on a specific architecture of hierarchical MDBS illustrated in figure 1.



**Fig. 1.** The architecture of Hierarchical MDBS.

The architecture of the transactions executed in the hierarchical MDBS is showed in figure 2. In this transaction hierarchical structure, top global transactions, denoted by  $G$ , are those transactions that should be executed in more than one sub-MDBS. A top global transaction is submitted to the top global transaction manager (TGTM) to schedule. TGTM divides the global transaction to a series of sub-transactions according to those sub-MDBSs who have data items that the global transaction need to access. In this paper, we constrain that every top global transaction can have at most one sub-transaction in each sub-MDBS to be scheduled. Local-global transactions, denoted by  $GL$ , are those transactions that want to access objects belonging to a single sub-MDBS but more than one local database. They are submitted to relative sub-MDBS transaction manager (LGTM) to schedule and do not need to let the TGTM know. LGTM divides transactions submitted to it to a series of sub-transactions according to those local databases who have data items that the transaction apply for. Those local transactions, denoted by  $TL$ , are submitted directly to the local transaction manager (LTM) to schedule and do not need to let the upper transaction manager know.



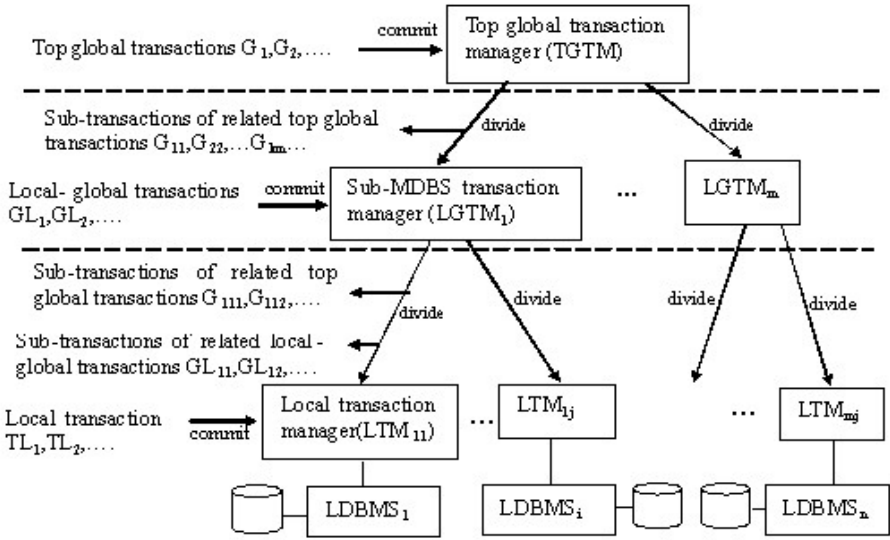


Fig. 2. The architecture of the transactions executed in hierarchical MDBS.

### 3 Conflict-Serializing Criterion in Hierarchical MDBS

#### 3.1 Preliminary

Note that all the discussions in the following part of this paper are based on the assumption that all the sub-MDBSs can ensure the serializable schedule of transactions submitted to them, and all the serializability of sub-schedules can be ensured by their local transaction managers. Another assumption of our discussion is that the hierarchical MDBS is a *failure-free* MDBS.

**[Definition 1]** *Global schedule  $S$* . Global schedule  $S$  is consist of the top global transaction schedule  $S_g$  and all the schedule  $S_k$  of  $GL$  in all sub-MDBS <sub>$k$</sub>  (for  $k=1, 2, \dots, m$ ). Denoted by  $S = S_g \cup (\cup_{k=1}^m S_k)$ . The schedule  $S_k$  in each sub-MDBS <sub>$k$</sub>  is just the same as that in traditional MDBS.

**[Theorem 1]** *Global serialization*. Let  $S$  be the schedule of top global transactions.  $S$  is serializable iff: all the sub-schedules  $S_k$  (for  $k=1, \dots, m$ .  $m$  is the amount of the sub-MDBS just below the top) of transactions in each sub-MDBS <sub>$k$</sub>  are serializable, and there is a total order  $O$  on global transactions in  $S$  such that for each sub-MDBS <sub>$k$</sub>  ( $1 \leq k \leq m$ ), the serialization order of global sub-transactions in  $S_k$  is consistent with  $O$  (it implies that all the schedule orders in LDBs are all consistent with  $O$  as well).

Theorem 1 tells us that if we can define a total order relationship among top global transactions that all schedules of sub-MDBS <sub>$k$</sub>  are consistent with, and find some ways to force the schedule of top global transactions to follow this total order, we can ensure the serializable schedule of top global transactions. In this paper, we extend the pre-version definition of the chain-conflict and make it meet the requirement of hierarchical MDBS environment.

**[Definition 2]** *Chain-conflicting transaction in hierarchical MDBS.*

- ① If the local transactions in  $LDB_i$  have a total order  $TL_{i1} \sim_c TL_{i2} \sim_c \dots \sim_c TL_{in}$ , these transactions are chain-conflicting.
- ② Suppose  $gl_k$  is the set of transactions that executed in sub-MDBS<sub>k</sub> (for  $k=1, \dots, m$ ), and  $TL_{kj}$  is the set of the transactions in  $gl_k$  which will be executed in local site  $LDB_j$  ( $1 \leq j \leq i_k$ ,  $i_k$  is the amount of local databases that constitute sub-MDBS<sub>k</sub>). If the sub-transactions in  $TL_{kj}$  have chain-conflict relationship and satisfy a total order  $O_k$  (that is satisfying ①), the transactions in  $gl_k$  are chain-conflicting.
- ③ Suppose  $g$  is the set of the top global transactions. If the sub-transactions of the transactions in  $g$  are chain-conflicting in each sub-MDBS<sub>k</sub> respectively and satisfy a total order  $O_k$  (that is satisfying ②), and all the  $O_k$  are the sub-order of a total order  $O$ , then the transactions in  $g$  are chain-conflicting in an order consistent with  $O$ .

(Note that ' $T_1 \sim_c T_2$ ' represents a conflict between  $T_1$  and  $T_2$ )

**[Definition 3]** *Chain-conflicting serializability in hierarchical MDBS.*

A schedule  $s$  is chain-conflicting serializable if the set  $T$  of committed transactions in  $s$  is chain-conflicting in a total order  $O$  on  $T$  and  $s$  is serializable in  $O$ .

### 3.2 Application of Extended Chain-Conflict in Hierarchical MDBS

Having introduced some concepts of chain-conflict in hierarchical MDBS, we now show how we can use it to the determination of a serializable top global schedule.

**[Theorem 2]** *Chain-conflicting serializability theorem* Let  $S$  be a global schedule of top global transactions, and  $g$  be the set of the top global transactions in  $S$ . If  $S_g$  is chain-conflicting serializable, and all the sub-schedules  $S_k$  (for  $k=1, \dots, m$ ) in sub-MDBS<sub>k</sub> are also chain-conflicting serializable, then  $S$  is serializable.

The proof of theorem 2 relies on one of the Lemma in reference [2].

**[Lemma]** If  $o_i$  and  $o_j$  are conflicting operations of transactions  $T_i$  and  $T_j$  (respectively) in a serializable schedule  $s$ , then  $o_i \prec_s o_j$  iff  $T_i \prec_{sr}^s T_j$ . (where ' $\prec_{sr}^s$ ' represents serial schedule  $s$ ).

**Proof of Theorem 2:** Suppose  $S_g$  is chain-conflicting serializable in a total order  $O$  ( $G_{i_1} \prec G_{i_2} \prec \dots \prec G_{i_n}$ ). Without loss of generality, we assume that, in sub-MDBS<sub>k</sub> (for  $1 \leq k \leq m$ ), sub-transaction of those top global transactions  $G_{i_{1k}}, G_{i_{2k}}, \dots, G_{i_{nk}}$  exist.

What we need to prove is that if  $S_k$  is serializable, then  $G_{i_{1k}} \prec_{sr}^{S_k} G_{i_{2k}} \prec_{sr}^{S_k} \dots \prec_{sr}^{S_k} G_{i_{nk}}$  holds. This proof proceeds by induction on a number  $n$  of top global transactions.

$n = 1$ : Straightforward.

Suppose for  $n=j(>1)$ ,  $G_{i_{1k}} \prec_{sr}^{S_k} G_{i_{2k}} \prec_{sr}^{S_k} \dots \prec_{sr}^{S_k} G_{i_{jk}}$  holds. Consider  $n=j+1$ . Because  $G_{i_{jk}}$  precedes  $G_{i_{j+1k}}$  in  $O$ , it means  $G_{i_{jk}} \sim_c G_{i_{j+1k}}$ . If  $o_{i_{jk}}$  and  $o_{i_{j+1k}}$  are conflicting operations of  $G_{i_{jk}}$  and  $G_{i_{j+1k}}$  respectively, then  $o_{i_{jk}} \prec_s o_{i_{j+1k}}$ . By Definition 2-③ and the

conflict operations exchangeable principle,  $o_{i_jk} \prec_{S_k} o_{i_{j+1}k}$  holds. Thus, by Lemma,  $G_{i_jk} \prec_{sr}^{S_k} G_{i_{j+1}k}$  also holds.

Thus, our induction proof shows that  $G_{i_1k} \prec_{sr}^{S_k} G_{i_2k} \prec_{sr}^{S_k} \dots \prec_{sr}^{S_k} G_{i_jk} \prec_{sr}^{S_k} G_{i_{j+1}k}$  holds. Hence, the serialization order of global sub-transactions in  $S_k$  is consistent with  $O$ . Consequently, by Theorem 1,  $S$  is serializable.

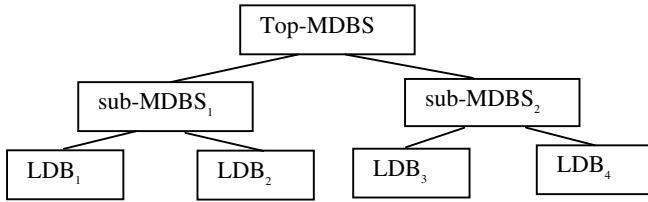
Now let's see an example of the application of the extended chain-conflicting.

**Example:** Consider the hierarchical MDBS in figure 3. The following three top global transactions are submitted to TGTm.

$$G_1: w_{G_1}(a) r_{G_1}(c) w_{G_1}(c) r_{G_1}(d)$$

$$G_2: r_{G_2}(a) r_{G_2}(d)$$

$$G_3: w_{G_3}(a) r_{G_3}(b) r_{G_3}(d)$$



**Fig. 3.** The hierarchical MDBS of the example

We suppose that the firstly submitted transaction will be scheduled first. Obviously, there are chain-conflicts  $G_{11} \sim_c G_{21} \sim_c G_{31}$  and  $G_{11} \sim_c G_{31}$  in sub-MDBS<sub>1</sub>, and no conflict in sub-MDBS<sub>2</sub>. Suppose TGTm schedule these three transactions in the order ' $G_1, G_2, G_3$ '. If a local-global transaction  $GL_{21}: w_{GL_{21}}(c) w_{GL_{21}}(d)$  exists in sub-MDBS<sub>2</sub> at the same time and is scheduled before other transactions, the schedules of the two sub-MDBSs may be:

$$S_1: w_{G_1}(a) r_{G_2}(a) w_{G_3}(a) r_{G_3}(b)$$

$$S_2: w_{GL_{21}}(c) r_{G_1}(c) w_{G_1}(c) r_{G_2}(d) w_{GL_{21}}(d) r_{G_1}(d) r_{G_3}(d)$$

Thus,  $S_1$  and  $S_2$  are both serializable schedule, but  $S$  is not. The conflict equivalent schedule of  $S_1$  is  $G_{11} \prec_{S_1} G_{21} \prec_{S_1} G_{31}$ . But the conflict equivalent schedule of  $S_2$  is  $G_{22} \prec_{S_2} GL_{21} \prec_{S_2} G_{12} \prec_{S_2} G_{32}$ . The reason is that there is no chain-conflict relationship of  $G_1, G_2, G_3$  in sub-MDBS<sub>2</sub> such that the LGTM of sub-MDBS<sub>2</sub> cannot control the execution order of them by chain-conflicting operation. If  $G_2$  had been changed as  $G_2: r_{G_2}(a) w_{G_2}(d)$ , a chain-conflict  $G_{12} \sim_c G_{22} \sim_c G_{32}$  would be formed in sub-MDBS<sub>2</sub>. According to the serializable criterion, there must exist  $r_{G_1}(d) \prec_{S_2} w_{G_2}(d)$  in  $S_2$ . Hence, the total order  $O \langle G_1, G_2, G_3 \rangle$  that both  $S_1$  and  $S_2$  are consistent with can be found in top global schedule  $S$ .  $S$  is chain-conflicting serializable.

For the convenience of implementation, we can apply Transaction Execution Graph to judge the chain-conflicting relation between top global transactions. In the graph, nodes represents the committed top global transactions, an edge like  $G_i \rightarrow G_j$

represents: a)  $G_i$  precedes  $G_j$  in  $O$ ; or b) there exists at least one (direct or indirect) conflict between  $G_{ik}$  and  $G_{jk}$  in sub-MDBS<sub>k</sub> (for  $1 < k < m$ ), and  $G_{ik} \prec_{S_k} G_{jk}$ , or  $G_{ik} \prec_{S_k} GL_{jk} \prec_{S_k} \dots GL_{jk} \prec_{S_k} G_{jk}$  exists. We can prove that if the graph is acyclic, the schedule is chain-conflicting serializable.

## 4 Conclusion and Further Discussion

The concepts of hierarchical MDBS show us a possible orientation for the integrity of large number of legacy database sources that locate in various distant sites and adopt different strategy of transaction management, such as the virtual enterprise application and the mobile database application. In this paper, an extended chain-conflicting serializability criterion for the correct schedule of transactions in hierarchical MDBS is addressed. The details of the criterion and its correctness and application are all given in this paper as well. The criterion can make the maintenance of the global transactions schedule more convenient. We do not need to care the transactions in local site, and just to control the chain-conflicting relation of top global transactions in a total order, then the serializability of global transactions schedule can be achieved. It is useful for those MDBS who have special requirement in autonomy of local member DBs.

Anyway, we have just studied the correctness criterion of hierarchical MDBS superficially, so flaws still exist. For example, our criterion may be too rigorous for some application and exert strong limitation on the concurrency control. It will lower the concurrency and the execution efficiency of transactions. However, to find a more relax and application-oriented criterion for the hierarchical MDBS environment is an important task in our future study. By the way, the effect of the failure should be taken into account and the solution of the problems should be studied in future work.

## References

1. Shared Mehrotra, Henry F.Korth, and A.Silberschatz. Concurrency Control in Hierarchical Multidatabase Systems. VLDB Journal [J], 1997, 6(2):152-172
2. Aidong Zhang, and Ahmed K.Elmagarmid. A Theory of Global Concurrency Control in Multidatabase Systems [J]. VLDB Journal, 1993,2(3): 331-360
3. W. Du and A. Elmagarmid, QSR: A Correctness Criterion for Global Concurrency Control in InterBase [C]. Proceedings of the 15th International Conference on VLDB, 1989
4. Shared Mehrotra, Rajeev Rastogi, Henry F.Korth, Ensuring Consistency in Multidatabases by Preserving Two-Level Serializability [J]. ACM Transactions on Database Systems, 1998,23(2):199-230
5. Wu Zhiqing, Lu Zhengding, Xiao Yibin. Approaches to Ensure the Correctness of Two Level Serialization (2LSR) Scheduling in MDBS. Journal of HUAZHONG University of Science and Technology [J], 2000, 28(9):18-20. (in Chinese)

# An Efficient Hierarchical Failure Recovery Algorithm Ensuring Semantic Atomicity for Workflow Applications\*

Yi Ren, Quanyuan Wu, Yan Jia, and Jianbo Guan

School of Computer Science,  
National University of Defence Technology, Changsha 410073, China  
renyi@nudt.edu.cn

**Abstract.** Failure recovery is essential for transactional workflow management systems. When a failure occurs, compensation is usually used to rollback corresponding processes and ensures semantic atomicity. Previously, many mechanisms have been proposed while performance problem is rarely considered. This is a limitation since compensating activities are rather expensive. In this paper, we propose an efficient hierarchical failure recovery algorithm, HFR. In stead of treating the compensation sphere as flat flow graphs, it generates compensation graph in a hierarchical manner from lower layers to higher ones according to nested structure of workflow execution history. The end compensation point is dynamically generated such that the compensation sphere is confined to a layer as low as possible and the number of compensations can be reduced. HFR can guarantee semantic atomicity of a workflow instance in case of a failure. The correctness of it is proved. Theoretical performance analysis shows that HFR has preferable efficiency.

## 1 Introduction

WfMS are used to coordinate and streamline the flow of processes and corresponding resources in distributed, heterogeneous and autonomous environment. These processes are supposed to guarantee transactional properties, such as atomicity. It is widely accepted that “all-or-nothing” requirement is too strict for transactional workflows. Relaxed transaction properties are required for long running, cooperative processes like workflows to avoid complete undo of performed work. Relaxed atomicity satisfying semantics of applications is called semantic atomicity [1].

Compensation is used to eliminate effects of executed sub processes and to rollback the workflow to a semantically acceptable state when failures occur. The concept was first introduced in Sagas [2]. It is nowadays generally considered a proper way to handle application rollback [3–9]. Though simple, compensation is very effective to simulate ACID properties for long-running database applications that would be too expensive to implement as ACID transactions.

It is typically complex to ensure semantic atomicity of workflow applications. First, identifying the compensation sphere, or the scope of the activities to be compensated, is a non-trivial task. A workflow process is often nested and complex struc-

---

\* The work has been co-supported by National Natural Science Foundation of China Under grant NO. 90104020 and China National Advanced Science & Technology (863) Plan under Contract No. 2001AA113020.

tured with complex control and data dependencies. Second, sub-processes of a workflow usually have different transactional behaviors. Third, a workflow application often performs intricate operations other than database reads and writes. So the activities are expensive to be compensated. And it is important to minimize the compensation sphere and avoid unnecessary efforts.

This paper proposes an algorithm, HFR (Hierarchical Failure Recovery), to handle semantic failures [10]. The contributions of this paper are summarized as follows. First, based on the conceptual model of workflows for semantic atomicity, we proposed notions of RM (Recovery Mode) and RP (Recovery Policy). According to their transactional behaviors, sub-processes and activities are specified with different modes and policies to improve the degree of automatic failure recovery capability. Second, HFR is proposed as a solution to minimize the compensation sphere. It calculates compensation graph hierarchically from lower levels to higher ones according to the nested structure of workflow execution history. At each level, it estimates if the workflow is recoverable by the predefined information and stops at the lowest recoverable level. Thus the compensation sphere is dynamically generated and number of compensation can be reduced. Theoretical performance analysis is presented. Third, many failure recovery mechanisms for transactional workflows have been introduced without correctness verified. We give the proof that HFR is correct for ensuring semantic atomicity of a workflow instance in case of a failure.

## 2 Related Work

Failure recovery of transactional workflows derives from the research of advanced transaction model (ATM) [2, 3, 11]. However, ATMs are typically database-centered and rigidly structured. Failure recovery of workflows is more complex for ATMs. Many process models have been described in [4–9, 12]. All assume the compensation sphere is somehow predefined and all the completed activities in the sphere are compensated. Even with partial compensation [5], the end compensation point (or the point where forward recovery starts) is pre-specified. This is often wasteful. In [8], a flexible mechanism for specifying the compensation sphere is proposed. It tries to reduce the number of activities that have to be compensated, but how to achieve semantic atomicity by compensation according to its proposed correct criteria is not mentioned. And the point where the compensation terminates is fixed. This may result in either unnecessary compensation or incorrect compensation. Our Work contrasts to previous work in that we aim at achieving lower costs by reducing the compensation sphere. And the end compensation point is determined dynamically.

## 3 Conceptual Workflow Model for Semantic Atomicity

A *workflow* is described as  $W = (G, O)$ , where  $G = (A, L)$  is a directed graph comprising a set of nodes  $A = \{a_1, a_2, \dots\}$  and the set of arcs  $L = \{l_1, l_2, \dots\}$  between nodes in  $A$ .  $\alpha_i$  refers to a non-dividable sub step of a workflow.  $\lambda_i = \langle a_j, a_k \rangle$  is the arc connecting  $a_j$  and  $a_k$ , which indicates the flow of control between them.  $O = \{o_1, o_2, \dots\}$  is a set of abstract data objects accessed and processed by nodes in set  $A$ . Start points

of  $W$  (denoted as  $S_W$ ) are nodes without incoming arcs, and end points of  $W$  (denoted as  $E_W$ ) are nodes without outgoing arcs. A workflow is required to have exactly one start point and at least one end point.

A *sub-process* of workflow  $W$  is described as  $P = (G', O')$ , where  $G' = (A', L')$  is a sub graph of  $G$  with one start node,  $A' \subseteq A$ ,  $L' \subseteq L$ ,  $O' \subseteq O$ . In the following sections, we denote start point of sub-process  $p$  as  $S_p$ . The adjacent higher process of a sub-process is called its parent process. An *activity*  $\alpha$  is a logical unit of work of workflow applications. It is actually a node in the workflow specification graph. A *block* is a sub-process with semantic atomicity requirement<sup>1</sup>. If the parent process  $p$  of block  $b$  is a block, then it is said that  $p$  is the parent block of  $b$ .

## 4 Hierarchical Failure Recovery

### 4.1 Recovery Mode and Recovery Policy

As mentioned in section 1, sub-processes of practical workflow applications typically have different transactional behaviors. So we introduce the concept Recovery Mode. Sub-processes with dissimilar RM are handled differently in case of failures.

**Definition 1 Recovery Mode (RM).** *RM* is the property which specifies atomicity feature of a sub-process  $p$ .  $RM(p) = \{atomic, non-atomic\}$ , where *atomic* means that  $p$  must execute as a semantic atom.

Sub-process  $p$  with  $RM(p) = atomic$  is actually a block. RM of an activity is always atomic. Similar to sub-processes, activities behave differently due to their dissimilar recovery feature in case of failures. Thus we introduce Recovery Policy of activities.

**Definition 2 Recovery Policy (RP).** *RP* is the property used to define recovery feature of activities in a block.  $RP(\alpha) = \{non-vital, compensable, critical\}$ , where policy *compensable* has its sub policies *retrievable*, *replaceable*.

An activity is *non-vital* if it is effect-free, i.e., its execution will not influence the results of all the other execution sequences of activities. An activity  $\alpha_i \in A$  is *compensable* if there is an activity  $\alpha_i^{-1} \in A$  where the sequence  $\langle \alpha_i, \alpha_i^{-1} \rangle$  is effect-free. Activity  $\alpha_i^{-1}$  is called the compensating activity of  $\alpha_i$ . Compensating activity semantically undoes the effects of the original activity. An activity  $a \in A$  is *critical* if it is neither *compensable* nor *non-vital*. It is not effect-free and its results cannot be eliminated. Obviously, failures in the same block after a completed critical activity cannot be handled automatically. Human intervention is needed.

For a *compensable* activity, its sub policy must be specified. A compensable activity is *retrievable* if it can finally execute successfully after being retried for finite times. And activity  $a \in A$  is *replaceable* if there is an activity  $a' \neq a$  and  $a'$  is functionally equivalent with  $a$ . Activity  $a'$  is called the replacement of  $a$ .

---

<sup>1</sup> All the activities are supposed to succeed or fail altogether. However, when a failure occurs, the sub-process is not assumed to rollback entirely. The sub-process may be compensated partially and continue by retrying or alternative methods. And it is acceptable as long as termination guarantees of the workflow application are satisfied.

Further, a sub-process  $p$  is said having an *alternative-path* if there is a sub-process  $p'$  where 1)  $p' \neq p$ , and 2)  $S_p = S_{p'}$ , and 3)  $p'$  and  $p$  is functionally equivalent, and 4)  $p$  has more than one node. Then  $p'$  is called an alternative-path of  $p$ . A sub-process  $p$  is called *non-recoverable* if it cannot recover automatically because of containing critical activities and it has no alternative-path.

It needs business level support to automate the failure recovery. Therefore, RM and RP must be pre-specified. And it is required to specify corresponding compensable activities and replacements. Sub-processes with alternative-paths should provide the definitions of their alternatives.

## 4.2 Algorithm Description

A failure occurs when there is an unexpected exception in an activity. For complex workflow with nested structures, a failure occurring in one layer needs to be propagated to a higher one until the workflow finally gets recovered or the administrator is notified. In this section, HFR is illustrated as four sub algorithms: *FailureForwarding*, *FlowRecovery*, *FAPreHandling* and *BCGGenerating*.

**(1) Failure forwarding.** *FailureForwarding* serves as the main algorithm of the four sub-algorithms. It takes the type of the exception and state parameters list as input, activates corresponding registered *ExceptionHandler* to handle the exception. If it is an unexpected type, forward it to *FlowRecovery*.

**(2) Flow recovery.** *FlowRecovery* is invoked by *FailureForwarding* to handle failures. It aborts executing activities and determines if the failure activity is recoverable by calling *FAPreHandling*. If recoverable, then reinitialize it based on the substituted definition generated by *FAPreHandling*, else goes to higher block (if there is) and calculates compensating graph of parent block by calling *BCGGenerating*. If the parent block is recoverable, the workflow is rolled back to start point and is restarted forward by executing alternative-path; else higher level is arranged and the sub-algorithm repeats until the workflow is recoverable or reaches the top block. If the top block is *non-recoverable*, rollbacks to its start point by corresponding compensating graph, else notifies the administrator for human intervention.

**Algorithm FlowRecovery** (Input: ID of the failure activity)

- 1) NonRecoverable := FALSE; Abort all the executing activities.
- 2) Initialize current compensating graph as NULL.
- 3) Invoke FAPreHandling, check return value of the NonRecoverable mark.
- 4) If it is FALSE, then initialize substituted definition of the activity and execute it. Return. Else set the failure activity as current failure block.
- 5) If current failure block has parent process  $p$  and RM( $p$ ) is atomic
  - 5.1) then call BCGGenerating, current compensating graph as input, get compensating graph of  $p$ . If  $p$  has alternative-path,
    - 5.1.1) then NonRecoverable := FALSE, rollback to  $S_p$  according to the compensating graph, reinitialize and execute the alternative path. Return.
    - 5.1.2) else set  $p$  as the current failure block, go to 5.
  - 5.2) Else rollback to the start point of the block, and notify the administrator. Wait for handling or return.



*FlowRecovery* provides backward recovery by compensation and supports forward recovery. It calculates the compensating graphs in a nested manner and merges the lower level graph to the higher ones. *FlowRecovery* does not terminate until the block can be recovered or it reaches the top block for manual recovery. Once a block is recoverable, it will not go to higher ones. Thus the start point of forward recovery or the end compensation point is determined dynamically during the bottom-up process. And the compensation sphere is restrained at a level as low as possible.

**(3) Pre-handling of failure activity.** *FAPreHandling* pre-handles failure activity for further processing. If it is non-vital, then redefine the activity as a null activity. If compensable, then rollback it. And the definition of failed replaceable activity will be substituted by its replacement for forward recovery. If *critical*, then set *NonRecoverable* as TRUE and return it.

**(4) Generation of compensating graph.** Compensating graph is useful in backward recovery. The goal of *BCGGenerating* is to generate compensating graph of current failure block's parent block according to workflow execution history (the trace of a workflow instance from the start point to the point where it terminates). Compensating graph of parent block is calculated based on that of current block.

**Algorithm BCGGenerating** (Input: current compensating graph; ID of current failure block; ID of current failure block's parent block; Output: compensating graph of the parent block)

- 1) Get workflow execution history graph  $G_H(A', E)$ .
- 2) Consider current failure block  $b$  as a node (called failure node).
- 3) Trace inversely from failure node to  $S_B$  in parent block  $B$ . Let  $N$  be the set of nodes passed by (including failure node).
- 4) Expand set  $N$ , such that  $N = N \cup \{a \mid (a \in A') \wedge (a \in \text{reachable}(N))\}$ .
- 5) Construct  $G_{H'}(A'', E')$ , maximal sub graph of  $G_H$ , where  $A'' = N$ , with connecting edges unchanged.
- 6) If compensating graph of  $b$ , i.e.,  $G_b$  is not NULL, then replace failure node in the  $G_{H'}$  with  $G_b$ , and edges keep unchanged.
- 7) Else if  $G_b$  is NULL, then delete the failure point in  $G_{H'}$  and related edges.
- 8) Reverse all the edges in  $G_{H'}$ .
- 9) If  $G_{H'}$  has multiple start points, then introduce a START node; add directed edge from START to the original start points.
- 10) Return  $G_{H'}$ , the compensating graph of  $B$ .

### 4.3 Discussion

HFR is based on the following assumptions:

1. Compensating activity is *retrievable* and alternative path can finally succeed.
2. Sub-processes of a block cannot be *non-atomic*.
3. If activity  $a$  is failure point in block  $b$  and there exists critical activity  $a_{cr}$  in the highest block  $b_{top}$  containing  $b$ , then: 1)  $a$  is the critical activity, or 2) From activity  $a$  to the upper layer, if there is a block  $b'$  with alternative path, then  $a_{cr} \prec S_{b'}$ , else  $a_{cr} = S_{b_{top}}$ .

According to assumption 2, RM of sub-processes of a block must be atomic. Actually the top block defines an atomic sphere. Note that critical activity is not *compensable*. If  $a_{cr}$  is between the start point of forward recovery (or end compensation point) and the failure point, then the effects of  $a_{cr}$  and that of the activities between  $a_{cr}$  and start point of forward recovery should be eliminated. This is impossible since  $a_{cr}$  is not compensable and not effect-free. Assumption 3 is to avoid this situation. It guarantees that failure recovery with the proposed algorithm will always finish with termination guarantees of workflows satisfied.

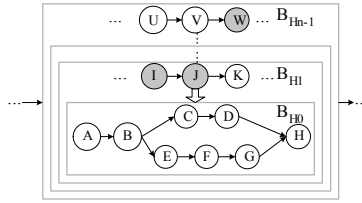


Fig. 1. Hierarchical structure of a block in a workflow specification

## 5 Correctness

According to the assumption 2, any sub-process  $p$  of a workflow instance  $w$  with  $RM(p)=atomic$  is structured as  $n$  layers of blocks. Let  $B_{H0}, B_{H1}, \dots, B_{Hn-1}$  denote the sets of blocks (including activities) corresponding to layers from lowest to the top one in the hierarchical structure shown in Fig.1, dark nodes are blocks, other nodes are activities. Elements of  $B_{Hi}$  ( $i=0, \dots, n-1$ ) are blocks and activities. Elements in  $B_{Hi+1}$  are often built up with those in  $B_{Hi}$ . For any activity defined in an atomic sphere of workflow instance  $w$ , it is in and only in one layer, says  $B_{Hi}$ . Theorem 1 and 2 are proved to illustrate correctness of HFR.

**Theorem 1** HFR can guarantee semantic atomicity of a workflow instance in case of a failure.

**Proof:** The failure of a block is caused by that of an activity. Therefore, we only need to analyze the correctness of failure handling caused by activities. For any failure activity  $a \in B_{Hi}$ :

- 1) If  $a$  is *non-vital*, then failure of  $a$  can be ignored. If  $a$  is *compensable*, then it is evident that the workflow is recoverable (see *FAPreHandling* and *FlowRecovery*).
- 2) If  $a$  is *critical*, then from *FlowRecovery*:
  - 2.1) If there is a recoverable parent block of  $a$ , then the workflow will be rolled back to start point of the parent block and execute forward.
  - 2.2) If  $a$  has parent block and it is *non-recoverable*, then if there is a block  $b \in B_{Hj}$ ,  $i < j \leq n-1$ , and  $b$  is recoverable, then it is same with the case 2.1, else if there is not a recoverable block until the top block is reached, then the sub-process will be rolled back to the point before execution of the atomic sphere, and human intervention is arranged.

Thereby, HFR makes a workflow rollback to start point of an atomic sphere when it is non-recoverable or recovers it in the scope of a sub atomic sphere, provided that the assumptions are satisfied.

**Theorem 2** HFR is capable of handling the failure of non-atomic sub-process properly.

**Proof:** It can be proved by using theorem 1. Omitted due to space constraints.

## 6 Performance Analysis

**Theorem 3** In a failure block with hierarchical structure, HFR confines the compensation sphere to a layer as low as possible.

This can be proved according to step 5 of *FlowRecovery*. Theorem 3 shows one of the advantages of HFR. When a failure occurs, the algorithm tries to prevent unnecessary compensations and reduce the compensation sphere. However, hierarchical manner of compensating graphs generation may spend more time than that of flat methods. So we compare costs of HFR with that of flat methods.

Assume that failure activity  $a$  is in an atomic sphere  $S$  with  $n$  layers (denoted as layer 0, ...,  $n-1$ ) and the block in layer  $i$  is recoverable. While corresponding blocks in lower layers are non-recoverable. Let the number of compensable activities in  $S$  be  $n_c$ , and total number of that from layer 0 to layer  $i$  is  $m_c$ . Mean time to compensate an activity is assumed as  $t_c$ , to access persistent compensating graph and execution history graph is  $t_{rwGH}$ , to generate compensating graph is  $t_{genGH}$ . And assume mean time for decision making at each layer be  $t_d$ .

(1) If the atomic sphere is non-recoverable, the number of compensating activities of HFR is equal to that of flat methods. But HFR accesses compensating graphs more often than the latter. The time difference of two methods for backward failure recovery is  $(n-2)*(t_{rwGH}+t_{genGH}+t_d)$ . As activities of complex workflow are often long running and compensating activities are expensive to compensate, the difference can be ignored as it is incomparable to executing time of compensating activities.

(2) If the atomic sphere is recoverable, then the time for HFR to execute backward recovery,  $t_{HFR}$ , is:

$$t_c * m_c + (i+1) * (t_{rwGH} + t_{genGH} + t_d)$$

And the time for flat methods,  $t_{cc}$ , is:

$$t_c * n_c + t_{rwGH} + t_{genGH} + t_d$$

So the difference is:

$$t_c * (n_c - m_c) + i * (t_{rwGH} + t_{genGH} + t_d)$$

Here,  $trwGH$  is actually the time accessing databases; it is much shorter compared to the time for compensating activities ( $t_c$ ). The time  $t_{genGH}$  and  $t_d$  are the same. Compared with  $trwGH$ ,  $t_{genGH}$  and  $t_p$ ,  $t_c$  is much longer. Thus the formula  $t_c \gg trwGH + t_{genGH} + t_d$  generally holds. So if failure block in the atomic sphere is recoverable, HFR is more efficient.

Above analysis shows that HFR is more efficient than flat failure recovery methods if it is recoverable in the scope of the atomic sphere. When the atomic sphere is non-

recoverable, the performance difference can be ignored. Since semantic failures of workflow applications are more often recoverable in the atomic sphere, HFR is preferable with regard to performance.

## 7 Conclusions

In this paper, we present HFR as a hierarchical failure recovery algorithm ensuring semantic atomicity for complex workflow applications. HFR can be used to avoid unnecessary compensation and to enhance the efficiency of failure recovery for complex transactional workflows. HFR has been applied to InforFlow system prototype developed at CIVC SE to enhance automatic capability of failure recovery.

## References

1. H. Garcia- Molina. Using Semantic Knowledge for Transaction Processing in a Distributed Database. *ACM Transactions on Database Systems*, June 1983, 8(2): 186-213
2. H. Garcia-Molina, et al. Sagas. In *proceedings of the ACM SIGMOD Conference on Management of Data*, San Francisco, California, USA (1987) 249–259
3. Zhang, A., et al. Ensuring Relaxed Atomicity for Flexible Transactions in Multidatabase Systems. In *proceedings of the ACM SIGMOD Conference*, ACM Press. Minneapolis, Minnesota, USA (1994) 67-78
4. C. Hagen and G. Alonso. Flexible Exception Handling in the OPERA Process Support System. In *Proceedings of the 18th International Conference on Distributed Computing Systems (ICDCS'98)*, IEEE Computer Society Press. Amsterdam, The Netherlands (1998) 526-533
5. P. Grefen, et al. Global transaction support for workflow management systems: from formal specification to practical implementation. *The VLDB Journal*, 2001, 10(4): 316-333
6. J. Eder, et al. The workflow activity model WAMO. In *proceedings of the Third International Conference on Cooperative Information Systems*. Vienna, Austria (1995) 87-98
7. F. Schwenkreis. A formal approach to synchronize long-lived computations. In *proceedings of the 5th Australasian Conference on Information Systems*. Melbourne, Australia (1994) 273-284
8. W Du, J Davis, MC Shan, U Dayal. Flexible Compensation for Workflow Processes. Technical Report HPL-96-72(R.1), HP Labs, available at <http://www.hpl.hp.com/techreports/96/HPL-96-72r1.pdf> (1997)
9. F. Leymann. Supporting Business Transactions via Partial Backward Recovery in Workflow Management Systems. In *Proceedings of BTW'95*, Springer Verlag. Dresden, Germany (1995) 51-70
10. G. Alonso, et al. Failure Handling in Large Scale Workflow Management Systems. Technical Report RJ9913, IBM Almaden Research Center, San Jose, CA (1994)
11. A. Elmagarmid (eds.). *Database Transaction Models for Advanced Applications*. Morgan Kaufmann Publishers (1992)
12. Q. Chen, U. Dayal. Failure Handling for Transaction Hierarchies. In *proceedings of 13th International Conference on Data Engineering*. Birmingham, UK (1997) 245–254

# A Macrocommittees Method of Combining Multistrategy Classifiers for Heterogeneous Ontology Matching\*

Leyun Pan, Hui Song, and Fanyuan Ma

Department of Computer Science and Engineering  
Shanghai Jiao Tong University, 200030 Shanghai, China  
pan-ly@cs.sjtu.edu.cn, {songhui\_17, fyyma}@sjtu.edu.cn

**Abstract.** To resolve the problem of ontology heterogeneity, we apply multiple classification methods to learn the matching between ontologies. We use the general statistic classification method to discover category features in data instances and use the first-order learning algorithm FOIL to exploit the semantic relations among data instances. When using multistrategy learning approach, a central problem is the combination of multiple match results. We find that the goal and the conditions of using multistrategy classifiers within ontology matching are different from the ones for general text classification. We propose a macrocommittees combination method that uses multistrategy in matching phase but not classification phase. In this paper we describe the combination rule called Best Outstanding Champion, which is suitable for heterogeneous ontology mapping. On the prediction results of individual methods, our method can well accumulate the correct matching of alone classifier.

## 1 Introduction

Bringing the meaning to the web data, in theory, ontology is a good solution for data interoperation. However, semantic heterogeneity or ontology heterogeneity is still a problem in real web environment. Because of the semantic web's distributed nature, ontologies from different communities will inevitably be different. The problem of improving system interoperability will rely on the reconciliation of different ontologies.

We can consider the process of addressing the semantic heterogeneity as the process of ontology matching (ontology mapping) [1]. Mapping processes typically involve analyzing the ontologies and comparing them to determine the correspondence among concepts. Given two ontologies in the same domain, we can find the most similar concept node in one ontology for each concept node in another ontology.

In our early work [2], we discussed the use of data instances associated with the ontology for addressing semantic heterogeneity. Our ontology matching system applies classification methods to learn the matching between the pair of ontologies. Given the source ontology B and the target ontology A, for each concept node in target ontol-

---

\* Research described in this paper is supported by Major International Cooperation Program of NSFC Grant 60221120145 and by Science & Technology Committee of Shanghai Municipality Key Project Grant 02DJ14045.

ogy A, we can find the most similar concept node from source ontology B. The system considers the ontology A and its data instances as the learning resource. All concept nodes in ontology A are the classification categories and relevant data instances of each concept are labeled learning samples in a classification process. The data instances of concept nodes in ontology B are unseen samples. The system classifies data instances of each node in ontology B into the categories of ontology A according to the classifiers for A. On the basis of classification results, the system computes the similarity degree of each pair of concept nodes using some similarity measure to find the matching relations between two ontologies.

Our system uses multistrategy, where each classifier exploits different type of information either in data instances or in the semantic relations among these data instances. We propose a macrocommittees combination method that uses multistrategy in matching phase but not classification phase. Using appropriate matching committee method, we can get better result than simple classifier.

This paper is organized as follows. In the next section, we will discuss the multiple classifiers for ontology matching. In section 3, we will discuss the different goal and the conditions of using multistrategy classifiers for ontology matching and propose the macrocommittees of matching and the evaluation method of classifiers for matching. Section 4 reviews related work. We give the conclusion in section 5.

## 2 Multiple Classifiers for Ontology Matching

The ontology matching system is trained to compare two ontologies and to find the correspondence among concept nodes. Our system uses multistrategy learning methods including both statistical and first-order learning techniques. Each base learner exploits well a certain type of information from the training instances to build matching hypotheses.

We use a statistical bag-of-words approach to classifying the pure text instances. The approach is the Naive Bayes learning technique, one of the most popular and effective text classification methods. It treats the textual content of each input instance as a bag of tokens, which is generated by parsing and stemming the words and symbols in the content. It is a kind of probabilistic models that ignore the words sequence and naively assumes that the presence of each word in a document is conditionally independent of all other words in the document.

Furthermore, the relations among concepts can help to learn the classifier. An appealing aspect of our approach is that the first-order rules can describe categories using a rich description of the local graph structure around the categories. Data instances under ontology are richly structured datasets, where data best described by a graph where the nodes in the graph are objects and the edges in the graph are links or relations between objects. The other methods for classifying data instances consider the words in a single node of the graph. However, the method can't learn models that take into account such features as the pattern of connectivity around a given instance, or the words occurring in instance of neighboring nodes. For example, we can learn a rule such as "An data instance belongs to *movie* if it contains the words *minute* and *release* and is linked to an instance that contains the word *birth*." Clearly, rules of this type, that are able to represent general characteristics of a graph, can be exploited to improve the predictive accuracy of the learned models.

This kind of rules can be concisely represented using a first-order representation. We can learn to classify text instance using a learner that is able to induce first-order rules. The learning algorithm that we use in our system is Quinlan's Foil algorithm [3]. Foil is a greedy covering algorithm for learning function-free Horn clauses definitions of a relation in terms of itself and other relations. Foil induces each Horn clause by beginning with an empty tail and using a hill-climbing search to add literals to the tail until the clause covers only positive instances.

Given the classification results, we can compute the degree of similarity using some similarity measure. Generally, the more data instances of one node in ontology B is classified into one node in ontology A, the more similar this pair of concept nodes is. The similarity measure can reflect this condition. [4] gives a detailed introduction into further similarity measures between two sets X and Y such as the matching coefficient  $\frac{|X \cap Y|}{|X| + |Y|}$ , the dice coefficient  $\frac{2|X \cap Y|}{|X| + |Y|}$ , the Jaccard or Tanimoto coefficient

or the overlap coefficient  $\frac{|X \cap Y|}{\min(|X|, |Y|)}$  is given. The final result of a clas-

sifier is the similarity matrix. The element of the matrix represents the similarity degree of a pair of concept nodes. We choose the Jaccard or Tanimoto coefficient measure in our system.

### 3 Micro-matching Committees and Evaluation of Matching Classifiers

Method of *Committees* (a.k.a. *ensembles*) is based on the idea that, given a task that requires expert knowledge to perform,  $k$  experts may be better than one if their individual judgments are appropriately combined. In text classification, the idea is to apply  $k$  different classifiers to the same task of deciding the category of a document, and then combine their outcomes appropriately. Various combination rules have been tested, such as *majority voting* (MV), *weighted linear combination* (WLC) and *dynamic classifier selection* (DCS) [5].

Similarly, we can use *matching committee* method in our ontology matching system. Though we use classification in our matching system, matching committee is different from classifier committee. For obtaining matching result, there are two different matching committee methods according to which phase uses multistrategy, in classification or matching:

- microcommittees: System firstly utilizes classifier committee. Classifier committee will negotiate for the category of each unseen data instance of source ontology. Then system will make matching decision on the base of single classification result. The multistrategy is used in classification phase.
- macrocommittees: System doesn't utilize classifier committee. Each classifier individually decides the category of each unseen data instance. Then system will negotiate for matching on the base of multiple classification results. The multistrategy is used in matching phase.

Let's see which committee is suitable for our ontology matching. To optimize the result of combination, generally, we wish we could give each member of committees

a weight reflecting the expected relative effectiveness of member. There are some differences between evaluations of classifiers in text classification and ontology matching.

In text classification, the initial corpus can be easily split into two sets, not necessarily of equal size: a *training(-and-validation) set*: The classifier is inductively built by observing the characteristics of these documents; a *test set*: used for testing the effectiveness of the classifiers. Each document in this set is fed to the classifier, and the classifier decisions are compared with the expert decisions. The trained classifier will be used to classify the unseen documents. Because the existence of validation set and test set, classifier committee can easily optimize the weight of members.

However, the boundary among training set, test set and unseen data instance set in ontology matching process is not obvious. These data sets have two features. Firstly, validation set and test set is absent in ontology matching process in which the instances of target ontology are regarded as training set and the instances of source ontology are regarded as unseen samples. Secondly, unseen data instances are not completely ‘unseen’, because instances of source ontology all have labels and we just don’t know what each label means. From the view with labels, unseen data instance set has little similarity with validation set and test set.

**Table 1.** Classifier 1

	$A'$
$A$	0.59
$B$	0.34
$C$	0.08
$D$	0.02

**Table 2.** Classifier 2

	$A'$
$A$	0.10
$B$	0.45
$C$	0.10
$D$	0.10

We adopt macrocommittees in our ontology matching system. Because of the first feature of data set, it is difficult to evaluate the classifiers in microcommittees. Microcommittees can only believe the prior experience and manually evaluate the classifier weights, as did in [1].

Notes that the second feature of data set, which the instances of source ontology have the relative “unseen” feature. When these instances are classified, the unit is not a single instance but a category. So we can observe the distribution of a category of instances in a classifier. Each classifier will find a champion that gains the maximal similarity degree in categories of target ontology. In these champions, some may have obvious predominance and the others may keep ahead other nodes just a little. Generally, the more outstanding one champions is, the more we believe it. For example, as shown in Table 1 and Table 2, there are two distribution vectors (0.59, 0.34, 0.08, 0.02) (0.10, 0.45, 0.10, 0.10). Though the similarity degree 0.59 between  $A$  and  $A'$  in classifier 1 is bigger than the similarity degree 0.45 between  $B$  and  $A'$  in classifier 2, we still argue that  $B$  is more appropriate than  $A$  because  $B$  is more outstanding in Classifier 2. Thus we can adopt the *degree of outstandingness* of candidate as the evaluation of effectiveness of each classifier. The degree of outstandingness can be observe from classification results and needn’t be adjusted and optimized on a validation set.



The degree of outstandingness can be measured according to the angle between distribution vector and coordinate axis. The more one champion is close to one of coordinate axes, the more outstanding it is. So we can normalize the distribution vector into a unit vector. The degree of outstandingness can be defined as the maximal component value of the unit distribution vector.

We propose a matching committee rule called the Best Outstanding Champion, which means that system chooses a final champion with maximal accumulated degree of outstandingness among champion-candidates. The method can be regarded as a weighted voting committee. Each classifier votes a ticket for the most similar node according to its judgment. However, each vote has different weight that can be measured by degree of champion's outstandingness. So given a source node and columns of classifiers' results under this source node, we can find a final champion for target nodes as follows: for each target node  $c$  and  $k$  classifiers, the pooled votes can be

calculated as  $Votes_c = \sum_{i=1}^k w_i v_i$ . Where  $Votes_c$  are combined votes of node  $c$ . The

component  $w_i$  is the maximal component value of the unit distribution vector in the classifier  $i$ . The component  $v_i$  equals 1 if node  $c$  is the champion in classifier  $i$ , otherwise 0.

## 4 Related Works

From perspective of ontology matching using data instance, some works are related to our system. GLUE [1] also uses multiple learning strategies to find matching between ontologies. Some strategies classify the data instances and another strategy Relaxation Labeler searches for the mapping configuration that best satisfies the given domain constraints and heuristic knowledge. However, automated text classification is the core of our system. We focus on the full mining of data instances for automated classification and ontology matching. By constructing the classification samples according to the feature property set and exploiting the classification features *in* or *among* data instances, we can furthest utilize the text classification methods. Furthermore, as regards the combination multiple learning strategies, [1] uses microcommittees and manually evaluate the classifier weights. But in our system, we adopt the degree of outstandingness as the weights of classifiers that can be computed from classification result. Not using any domain and heuristic knowledge, our system can automatically achieve the similar matching accuracy. [6] computes the similarity between two taxonomic nodes based on their TF/IDF vectors, which are computed from the data instances. The method is so simple that can't satisfy the application of the real-world domain.

Some related works [7,8] also compare ontologies using similarity measures, whereas they compute the similarity between lexical entries. [9] describe the use of FOIL algorithm in text classification and information extraction for constructing knowledge bases from the web.

## 5 Conclusion

When current web evolves to semantic web, the data heterogeneity will transfer to the semantic heterogeneity between ontologies. We propose a multistrategy learning approach for resolving ontology heterogeneity on semantic web. In the paper, we apply the classification method to sum up the number of data instances belonging to a concept node pair. We use the general statistic classification method to discover category features in data instances and use the first-order learning algorithm FOIL to exploit the semantic relations among data instances. We indicate the similarities and differences between classifier committees and matching committees and propose the macrocommittees combination method. The system combines their outcomes using our matching committee rule called the Best Outstanding Champion, which is suitable for heterogeneous ontology mapping. The method can well accumulate the correct matching of alone classifier.

## References

1. A. Doan, J. Madhavan, P. Domingos, and A. Halevy. Learning to Map between Ontologies on the Semantic Web. In *Proceedings of the World Wide Web Conference (WWW-2002)*.
2. Leyun. Pan, Shui.Yu, Fanyuan. Ma. SIMON: A Multi-Strategy Classification Approach Resolving Ontology Heterogeneity On the Semantic Web. The Sixth Asia Pacific Web Conference 2004.
3. J. R. Quinlan, R. M. Cameron-Jones. FOIL: A midterm report. In *Proceedings of the European Conference on Machine Learning*, pages 3-20, Vienna, Austria, 1993.
4. C.D. Manning and H. Schuetze. *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, Massachusetts, 1999.
5. F. Sebastiani. Machine Learning in Automated Text Categorization. *ACM Computing Surveys*, Vol. 34, No. 1, March 2002.
6. M. Andrea Rodríguez, Max J. Egenhofer. Determining Semantic Similarity Among Entity Classes from Different Ontologies. *IEEE Transactions on Knowledge and Data Engineering*, (in press).
7. A. Maedche, S. Staab. Comparing Ontologies- Similarity Measures and a Comparison Study. Internal Report No. 408, Institute AIFB, University of Karlsruhe, March 2001.
8. M. Lacher, G. Groh. Facilitating the exchange of explicit knowledge through ontology mappings. In *Proceedings of the 14th Int. FLAIRS conference*, 2001.
9. M.Craven, D. DiPasquo, D. Freitag, A. McCalluma, T. Mitchell. Learning to Construct Knowledge Bases from the World Wide Web. *Artificial Intelligence*, Elsevier, 1999.

# Hypertext Classification Algorithm Based on Co-weighting Multi-information

Ya Peng, Ya-ping Lin, and Zhi-ping Chen

Computer and Communication College, Hunan University  
Hunan Changsha, 410082, China  
freiya@21cn.com, {yplin,jt\_zpchen}@hnu.cn

**Abstract.** Compared with the text information text, hypertext information such as hyperlinks and meta data all provide rich information for classifying hypertext documents. After analyzing different rules of using hypertext, we present a new hypertext classification algorithm based on co-weighting multi-information. We co-operate different hypertext information generally, by co-weighting them after extraction. Experimental results on two different data sets show that the new algorithm performs better than using single hypertext information individually.

## 1 Introduction

As the size of the Web expands rapidly, the need for automated hypertext classification techniques has become more apparent. A recent user[CH] study shows that users often prefer navigating through directories of pre-classified content. Automated hypertext classification poses new research challenges because of the rich information in a hypertext document and the connectivity among documents. Hyperlinks, HTML tags, and meta data all provide rich information for hypertext classification. Researchers have only recently begun to explore the issues of exploiting hypertext information for automated classification.

Chakrabarti [SC] studied the use of citations in the classification of IBM patents where the citations between documents (patents) were considered as “hyperlinks” and the categories were defined on a topical hierarchy. They obtained a 31% error reduction.

Joachims [JO] focused on Support Vector Machines (SVMs) with different kernel functions. Using one kernel to represent a document based on its local words, and another one to represent hyperlinks. Their experiments suggest that the kernels can make more use of the training-set category labels.

Yang[YA] presents a list of some simple kinds of hypertext regularities. However, the methods of integrating of those regularities and information are not provided.

The work summarized above, provides initial insights to exploit information in hypertext documents for automated classification, many questions still remain unanswered. Different rules and characteristics are not synthesized; using a specific regularity of hypertext, for example, will decrease classification accuracy to some data sets, but increase the accuracy to others[YA]. We find that

using a single regularity of hypertext, can't integrate the information all sided. In this paper, we synthesize different regularities of using hypertext information, and present a Hypertext Classification Algorithm Based on Co-weighting Multi-information. Before classification, this algorithm extracts title, hyperlinks and tags etc. from hypertext documents, and co-weights these information in classification. We co-operate different hypertext information generally, by co-weighting them after extraction. Experimental results on two different data sets with different traditional classifiers (NB[MC], TFIDF[JO2] and KNN[YA2] show that the new algorithm performs better than using single hypertext information individually.

## 2 Classification Algorithms

In this paper we improve three traditional classification algorithms. So we first introduce the three traditional classification algorithms briefly.

(1) *Naive Bayes Algorithm* NB algorithm count the pre-probabilities  $P(w_t|c_j)$  firstly, the probability of individual word  $w_t$  in vocabulary list  $V$ , given the document class  $c_j$ ; then count the pre-probability of each class  $P(c_j)$ :

$$P(w_t|c_j) = \frac{1 + \sum_{i=1}^{|D|} N(w_t, d_i) P(c_j|d_i)}{|V| + \sum_{s=1}^{|V|} \sum_{i=1}^{|D|} N(w_s, d_i) P(c_j|d_i)} \quad (1)$$

The counting formula of  $P(c_j)$  is :

$$P(c_j) = \frac{1 + \sum_{i=1}^{|D|} P(c_j|d_i)}{|C| + |D|} \quad (2)$$

(2) *TFIDF Algorithm* This algorithm regards each word as feature item  $t_k$ , each document as a vector  $\vec{d}_i = (w_{i1}, w_{i2}, \dots, w_{in})$  composed by words.

$$w_{ik} = t f_{ik} * i d f_k \quad (3)$$

$$i d f_k = \log(N / d f_k + 1) \quad (4)$$

The algorithm adds up all the document vector, and gets each class vector  $\vec{c}_i = (w_{i1}, w_{i2}, \dots, w_{in})$ . This algorithm counts the similarity of document vector and class vector  $sim(\vec{d}_i, \vec{c}_j)$ . And the most similar class  $c_j$  with the highest similarity value, is tested document's class.

$$\vec{c}_j = \sum_{d \in c_j} \vec{d}_i \quad (5)$$

$$class(d_i) = \arg \max_{c_j \in C} sim(\vec{d}_i, \vec{c}_j) \quad (6)$$

$$sim(\vec{d}_i, \vec{c}_j) = \frac{\vec{d}_i \cdot \vec{c}_j}{\|\vec{d}_i\| \cdot \|\vec{c}_j\|} = \frac{\sum_{k=1}^n w_{ik} * w_{jk}}{\sqrt{\sum_{k=1}^n w_{ik}^2 * \sum_{k=1}^n w_{jk}^2}} \quad (7)$$

(3)*KNN Algorithm* The idea of KNN algorithm: Given a test document, search for the  $k$  nearest neighbors in training set, judge the class which these  $k$  neighbors most belong to, and set this document class with it.

### 3 Hypertext Classification Algorithm Based on Co-weighting Multi-information

In this paper, we present a new hypertext classification algorithm based on co-weighting multi-information. We summarize the recent work about hypertext. So first some typical regularities of using hypertext is introduced.

#### 3.1 Regularities of Using Hypertext information

There are many methods of using hypertext information for classification. Some typical hypertext regularities and experimental analyses are used in this paper, which provides basic theory support for the new algorithm.

1. No Hypertext Regularity (page only). It is important to be aware that for some hypertext corpora and classification tasks on them, the only useful place is the document itself.
2. Pre-classified Regularity. There are regularities in the hyperlink structure itself. One such regularity prevalent on the web consists of a single document, which contains hyperlinks to documents, which share the same topic. We call this pre-classified regularity.
3. Meta Data Regularity. For many classification tasks that are of practical and commercial importance, meta data are often available from external sources on the web that can be exploited in the form of additional features, which is very useful for classification.

#### 3.2 Main Idea and Algorithm Description

Hypertext classification algorithm based on co-weighting multi-information builds a statistical model by synthesizing different regularities of using hypertext, and weights the title, hyperlinks, and HTML tags etc. extracted from the hypertext documents of data set. After the processing of weighting, it integrates the structured multi-information and uses the classifier for classification.

The details of the algorithm is described as follows:

1. Input the hypertext data set;
2. Preprocess the documents of data set. Extract the hypertext feature vector (hypertext information), words in the hyperlinks (pre-classification), HTML titles (meta information), words on the page (page only), and etc. We mark them with  $L_{1i}[t_1, t_2, \dots, t_{|L_1|}], \dots, L_{ni}[t_1, t_2, \dots, t_{|L_n|}]$ . Where,  $L$  is the feature vector of hypertext,  $i$  is the  $i$ th document,  $t$  indicates the word in the vector. For example,  $L_{xi}$  indicates the feature of HTML title in the  $i$ th document, where,  $t$  indicates the word in the HTML title.

3. Build hypertext classifier. During the phase of training and modeling for classifier, such as TFIDF. Through scanning words in the word frequency matrix occurring in the hypertext feature vector, weighting the word with  $\omega$ .  $\omega_x$  is the weight to the vector  $L_x$ ,  $w \rightarrow w * \omega$ . Assuming the sum of all feature item weights is equal to 1,  $\omega_1 \omega_2 \dots \omega_n = 1$ . The weight indicates the importance of hypertext feature item for classification. The detailed steps is as followed:
  - (a) Scan the training documents, build the word frequency matrix;
  - (b) Based on the word frequency statistical matrix, according to the selected algorithm, the corresponding weights of word vector in training documents are calculated.
  - (c) Through scanning the frequency of word in word frequency matrix occurring in the hypertext feature vector, we multiply the weight with the weighting modulus. If the word  $w_{ik}$  occurs in  $m$  different feature items,  $0 \leq m \leq n$ , then  $\omega_{ik}$  is the sum of  $m$  different feature items with weights.
4. Classify the test hypertext document set. To a new unclassified hypertext document, we preprocess it and use the established classifier model to weight the word vector of text document.

## 4 Experiments and Analysis

The experiment in this paper uses two hypertext data sets. One is WebKB [MC2] dataset. Besides, we collect 1500 corporation web pages from the web site “www.Hoovers.com”, which cover 5 classes, such as aviation, and each class includes 300 web pages. (“hoovers-5”). This paper takes the popular text classification system “Rainbow” [MC2] as experimental flat. The “Rainbow” system is developed in American CMU. We insert the code of co-operate training (COT) algorithm, compile and execute it in the environment of Linux. We respectively combine three traditional classifiers: NB, KNN and TFIDF with hypertext feature vector to build three corresponding hypertext classifiers, test the performance in experiments, compare and analyze the results.

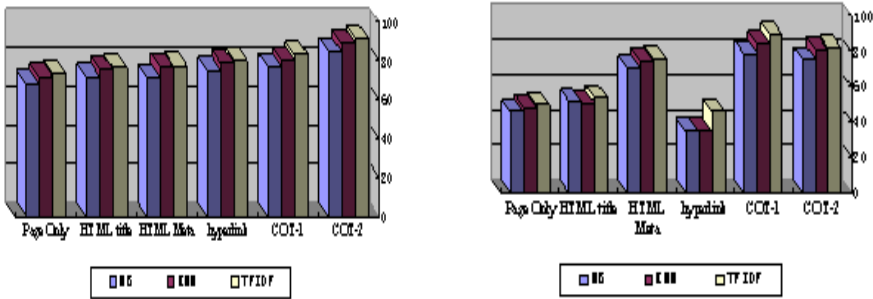
The experiments in this paper include two parts:

1. Firstly, we make use of different hypertext information for classification. The hypertext information includes words on the web pages (“page only” rule), HTML title, HTML Meta (“meta information” rule), and words in the hyperlink (“pre-classification” rule). In the experiment, we take 90% of the data set for training, and set aside 10% for testing. Experimental results are showed on Table 1.  
Table 1 shows that, for data set WebKB, the accuracy of classification using hypertext information is higher than that of without using hypertext information, but there are still some exceptions.
2. In the second part of the experiment, we extract hypertext information to form feature vector with weighting, and build a new classifier based on COT. We assume the weighting modulus of hypertext feature as follows: page words

**Table 1.** Classification accuracy of using different hypertext info.

	WebKB			Hoovers-5		
	NB	KNN	TFIDF	NB	KNN	TFIDF
words on pages	68.3	72.5	74.0	45.8	48.0	49.6
HTML title	73.2	77.0	79.5	51.5	50.3	53.2
HTML Meta	72.8	78.5	77.3	70.4	74.5	75.4
Hyperlink words	76.4	80.0	82.0	40.0	35.0	45.8

$\omega_1$ , HTML title  $\omega_2$ , HTML Meta  $\omega_3$ , Hyperlink words  $\omega_4$ . Based on the analysis of the first part of experimental results, we assign higher weight value to the hypertext information that behaves better individually. Through repeatedly testing and adjusting, we assume two kind of weighting modulus:  $\omega_1=0.1$ ,  $\omega_2=0.2$ ,  $\omega_3=0.4$ , and  $\omega_4=0.3$  (shown as COT-1);  $\omega_1=0.2$ ,  $\omega_2=0.2$ ,  $\omega_3=0.2$ , and  $\omega_4=0.4$  (shown as COT-2). Fig.1 shows the results.



(a) WebKB Dataset

(b) Hoovers-5 Dataset

**Fig. 1.** Classification Accuracy of COT and other individual regularities on 3 datasets.

Fig.1. shows the performance of hypertext classification using only single hypertext information and co-weighting information for two data sets. Firstly, we can note that for different data sets and different classifiers, the performance of co-weighting multi-information classification is better than only using single hypertext information generally. It suggest that COT can make full use of hypertext information resources and has good stability compared to the advantages of using single features. Secondly, we can note that the two new classifiers behave inversely for two datasets.

## 5 Conclusions

Hypertext documents provide rich information for classification. As the variety of hypertext information, using a single regularity of hypertext, can't integrate

the information all sided, and influent classification performance unsteadily. In this paper, we synthesize different regularities of using hypertext information and present a Hypertext Classification Algorithm Based on Co-weighting Multi-information. This algorithm extracts title and hyperlinks etc. from hypertext documents, and co-weights these information in classification. In this way, it integrates different hypertext information generally. Experimental results on three different data sets with different classifiers show that the new algorithm performs better than using single hypertext information individually. In this paper, we cooperate different regularities of hypertext by co-weighting the related information linearly, and set the weighting coefficients by analyzing the experiments on different regularities individually. As the weighting coefficients are set tentatively, we can go further study with the method of co-weighting.

## References

- [CH] H. Chen. Bringing order to the web: Automatically categorizing search results. In *Proceeding of CHI'00, Human Factors in Computing Systems* (2000)
- [JO] T. Joachims. Composite kernals for hypertext categorization. In *International Conference on Machine Learning(ICML'01)*, San Francisco, CA (2001) Morgan Kaufmann.
- [JO2] Joachims,T. A probabilistic Analysis of the Rocchio Algorithm with TFIDF for Text Categorization. *Machine Learning: Proceedings of the Fourteenth International Conference*(1997) 143–151
- [MC] McCallum, A. ,&Nigam, K. A Comparison of Event Model for Navie Bayes Text Classification. In *AAAI-98 Workshop on Learning for Text Categorization of the Fifteenth International Conference(ICML'98)* (1998) 59–367
- [MC2] McCallum.”Bow: A toolkit for statistical language modeling,text retrieval,classification and clustering.” <http://www.cs.cmu.edu/~mccallum/bow> (1996)
- [SC] Soumen Chakrabarti. Enhanced hypertext categorization using hyperlinks.*Proceedings ACM SIGMOD International Conference on Management of Data*,x Seattle, Washington, June ,ACM Press (1998) 307–318
- [YA] Yiming Yang, Sean Slattery, and Rayid Ghani. A study of approaches to hypertext categorization. *Journal of Intelligent Information Systems*. 18(2/3) (2002) 219–241
- [YA2] Yang,Y. An example-based mapping method for text classification and retrieval. *ACM Transactions on Information Systems* 23(3)(1994) 252–277



# Verifying a MAC-Based Service Bundle Authentication Mechanism for the OSGi Service Platform

Young-Gab Kim<sup>1</sup>, Dongwon Jeong<sup>2</sup>, and Doo-Kwon Baik<sup>1</sup>

<sup>1</sup> Dept. of Computer Science & Engineering, Korea University,  
1, 5-ka, Anam-dong, Sungbuk-gu, Seoul, 136-701, Korea  
{ygkim, baik}@software.korea.ac.kr

<sup>2</sup> Research Institute of Information and Communication Technology, Korea University,  
1, 5-ka, Anam-dong, Sungbuk-gu, Seoul, 136-701, Korea  
withimp@korea.ac.kr

**Abstract.** In this paper, we propose a service bundle authentication mechanism considering characteristics for the home gateway environment. We also designed the key exchange mechanism for exchanging a key in bootstrapping step. Furthermore we verify the safety of the key exchange mechanism and service bundle authentication mechanism with BAN Logic. Through the verification, we are able to show that the result does not compromise the security of the proposed service bundle authentication mechanism.

## 1 Introduction

In previous work [1,2], a service bundle authentication [3] mechanism based on MAC [4] has been proposed. The proposed mechanism is more efficient than PKI-based [5] service bundle authentication or RSH protocol [6] in the OSGi platform [7]. In order to check the security of the MAC-based service bundle authentication mechanism, we adopted the widely used BAN logic [8,9,10,11] to verify the proposed authentication mechanism.

This paper is organized as follows: Chapter 2 briefly presents the BAN logic, and Chapter 3 describes the verification procedure of the proposed authentication mechanism. Chapter 4 concludes the paper. In this paper, we more focus on verification of a service bundle authentication mechanism in OSGi Service platform than explanation of that. To get the details about description of the bundle authentication mechanism, please refer to [1,2].

## 2 The BAN Logic

Authentication protocols are typically described by listing the messages sent between the principals, and by symbolically showing the source, the destination and the contents of each message. This conventional notation is not convenient for manipulation in logic since the contents of each message often means more than necessary in the

security sense. BAN logic is widely used logic specifically designed for analyzing the protocols for the authentication of principals in distributed computing systems. With the BAN notations, we can transform each message into a logical formula, which is an idealized version of the original message. So we use the BAN logic to verify and analyze of the proposed service bundle mechanism in this paper. We use the BAN logic with four description parts: Usual notation, idealized protocol, assumption and goal. To analyze a protocol, we must first generate idealized protocol from the original one with the BAN logic. Assumptions about initial state are written and then logical formulas are attached to the protocol statements. The logical postulates are applied to the assumptions and the assertions, in order to discover the beliefs held by parties in the protocol. This procedure is repeated through all the protocol messages. To get the details about description of basic notation and logical postulates of BAN logic, please refer to [8,11]

### 3 Verification of the Key Exchange Mechanism and the Service Bundle Authentication Mechanism

#### 3.1 Verification of a Key Exchange Mechanism

As shown in Table 1, we first describe the proposed key exchange mechanism with usual notation. Here, message 1 denotes steps from (1) to (2) and message 2 denotes steps from (3) to (5) in Fig. 2 of [1]. Message 3 denotes steps from (6) to (9) in Fig.2 of [1].

**Table 1.** Usual notation of key exchange mechanism using a symmetric-key

□ Usual Notation	
Message 1	$SG \rightarrow OP : ID_{sg}, ID_{op}, \{T_{sg}, N_{sg}\}_{K_{sg\_op}}$
Message 2	$OP \rightarrow SG : ID_{sg}, ID_{op}, \{T_{sg+1}, T_{op}, N_{sg}, N_{op}\}_{K_{sg\_op}}$
Message 3	$SG \rightarrow OP : ID_{sg}, ID_{op}, \{T_{op+1}, N_{op}\}_{K_{sg\_op}}$

The idealized protocol can be presented using BAN logic as follows. (In this idealized form, we omit parts of the message that do not contribute to the beliefs of the recipient.) :

**Table 2.** Idealized protocol of key exchange mechanism using a symmetric-key

□ BAN Idealization	
Message 1	$SG \rightarrow OP : \{T_{sg}, N_{sg}\}_{K_{sg\_op}}$
Message 2	$OP \rightarrow SG : \{ \langle T_{sg+1}, SG \stackrel{N_{op}}{\rightleftharpoons} OP \rangle N_{sg} \}_{K_{sg\_op}}$
Message 3	$SG \rightarrow OP : \{ \langle T_{op+1}, SG \stackrel{N_{sg}}{\rightleftharpoons} OP, OP \stackrel{N_{op}}{\rightleftharpoons} SG \rangle N_{op} \}_{K_{sg\_op}}$

As shown in Table 3, in order to verify the idealized protocol, some assumptions are needed as follows.

**Table 3.** Assumptions of key exchange mechanism using a symmetric-key

□ BAN Assumption			
(1) $SG \models SG \stackrel{K_{SG,OP}}{\leftrightarrow} OP$ ,	(2) $OP \models SG \stackrel{K_{SG,OP}}{\leftrightarrow} OP$ ,	(3) $SG \models SG \stackrel{N_{SG}}{\leftrightarrow} OP$ ,	(4) $OP \models SG \stackrel{N_{OP}}{\leftrightarrow} OP$
(5) $SG \models \#(N_{SG})$ ,	(6) $OP \models \#(N_{OP})$ ,	(7) $SG \models \#(T_{SG})$ ,	(8) $OP \models \#(T_{OP})$
(9) $SG \models \#(T_{OP})$ ,	(10) $OP \models \#(T_{SG})$		

We premise that only the embedded service bundle can access symmetric-key in the service gateway, and the operator is protected by a firewall. Furthermore both the service gateway and the operator have a random generator. The final goal of the key exchange system is to create a shared secret key between the service gateway and the operator with a mutual authentication and it is described with BAN logic as follows in Table 4.

**Table 4.** Goal of key exchange mechanism using symmetric-key

□ BAN Goal			
(1) $OP \models \#(SG \stackrel{N_{SG}}{\leftrightarrow} OP)$	(2) $SG \models \#(SG \stackrel{N_{OP}}{\leftrightarrow} OP)$		
(3) $OP \models SG \models SG \stackrel{N_{SG}}{\leftrightarrow} OP$	(4) $SG \models OP \models SG \stackrel{N_{OP}}{\leftrightarrow} OP$		

Table 5 and 6 show analysis and verification of the key exchange mechanism using BAN logic.

**Table 5.** Verification of key exchange mechanism using a symmetric-key(In message 2)

<b>In Message 2</b>	
<b>STEP 1</b>	
$SG \models SG \stackrel{K_{SG,OP}}{\leftrightarrow} OP, SG \vdash \{ \langle T_{SG}+1, SG \stackrel{N_{OP}}{\leftrightarrow} OP \rangle N_{SG} \} K_{SG,OP}$	
$SG \vdash \langle T_{SG}+1, SG \stackrel{N_{OP}}{\leftrightarrow} OP \rangle N_{SG}$	
$SG \vdash (T_{SG}+1, SG \stackrel{N_{OP}}{\leftrightarrow} OP)$	
<b>STEP 2</b>	
$SG \models \#(T_{SG}+1)$	
$SG \models \#(T_{SG}+1, SG \stackrel{N_{OP}}{\leftrightarrow} OP)$	
<b>STEP 3</b>	
$SG \models \#(T_{SG}+1), SG \models \#(T_{SG}+1, SG \stackrel{N_{OP}}{\leftrightarrow} OP)$	
$SG \models \#(SG \stackrel{N_{OP}}{\leftrightarrow} OP)$	
<b>STEP 4</b>	
$SG \models SG \stackrel{N_{SG}}{\leftrightarrow} OP, SG \vdash \langle T_{SG}+1, SG \stackrel{N_{OP}}{\leftrightarrow} OP \rangle N_{SG}$	
$SG \models OP \vdash (T_{SG}+1, SG \stackrel{N_{OP}}{\leftrightarrow} OP)$	
$SG \models OP \vdash SG \stackrel{N_{OP}}{\leftrightarrow} OP$	
<b>STEP 5</b>	
$SG \models \#(SG \stackrel{N_{OP}}{\leftrightarrow} OP), SG \models OP \vdash SG \stackrel{N_{OP}}{\leftrightarrow} OP$	
$SG \models OP \models SG \stackrel{N_{OP}}{\leftrightarrow} OP$	

The goal of (2) and (4) in Table 4 is achieved through 5 steps in message 2. It denotes that the service gateway authenticates the operator by comparing the  $N_{SG}$ , which the service gateway sent to the operator, with the  $N_{SG}$ , which the operator sent to the service gateway. The next authentication process in message 3 is as follows:

**Table 6.** Verification of key exchange mechanism using symmetric-key(In message 3)

<b>In Message 3</b>	
<b>STEP 1</b>	
$OP \models SG \xleftrightarrow{K_{SG,OP}} OP, OP \nVdash \{ \langle T_{OP}+1, SG \xleftrightarrow{N_{SG}} OP, OP \models SG \xleftrightarrow{N_{OP}} OP \rangle \}_{K_{SG,OP}}$	
$OP \nVdash \langle T_{OP}+1, SG \xleftrightarrow{N_{SG}} OP, OP \models SG \xleftrightarrow{N_{OP}} OP \rangle$	
$OP \nVdash (T_{OP}+1, SG \xleftrightarrow{N_{SG}} OP, OP \models SG \xleftrightarrow{N_{OP}} OP)$	
<b>STEP 2</b>	
$OP \models \#(T_{OP}+1)$	
$OP \models \#(T_{OP}+1, SG \xleftrightarrow{N_{SG}} OP, OP \models SG \xleftrightarrow{N_{OP}} OP)$	
<b>STEP 3</b>	
$OP \models \#(T_{OP}+1), OP \models \#(T_{OP}+1, SG \xleftrightarrow{N_{SG}} OP, OP \models SG \xleftrightarrow{N_{OP}} OP)$	
$OP \models \#(SG \xleftrightarrow{N_{SG}} OP)$	
<b>STEP 4</b>	
$OP \models SG \xleftrightarrow{N_{OP}} OP, OP \nVdash \langle T_{OP}+1, SG \xleftrightarrow{N_{SG}} OP, OP \models SG \xleftrightarrow{N_{OP}} OP \rangle$	
$OP \models SG \vdash (T_{OP}+1, SG \xleftrightarrow{N_{SG}} OP, OP \models SG \xleftrightarrow{N_{OP}} OP)$	
$OP \models SG \vdash SG \xleftrightarrow{N_{SG}} OP$	
<b>STEP 5</b>	
$OP \models \#(SG \xleftrightarrow{N_{SG}} OP), OP \models SG \vdash SG \xleftrightarrow{N_{SG}} OP$	
$OP \models SG \models SG \xleftrightarrow{N_{SG}} OP$	

The goal of (1) and (3) in Table 4 is achieved via 5 steps in message 3. This means that the operator authenticated the service gateway by comparing the  $N_{OP}$ , which the operator sent to the service gateway, with the  $N_{OP}$ , which the service gateway sent to the operator. Therefore the mutual authentication is accomplished by the service gateway authenticating the operator in step 5 in message 2, and the operator authenticating the service gateway in step 5 in message 3. After this authentication step, the operator and the service gateway generate a shared secret, which is necessary to create MAC value using the  $N_{SG}$  and the  $N_{OP}$ .

### 3.2 Verification of a Service Bundle Authentication Mechanism

A usual notation of MAC-Based Service Bundle Authentication Mechanism is described in Table 7. Message 1 denotes steps from (1) to (2) in Fig.4 of [1], and Message 2 denotes steps from (3) to (5) in Fig.4 of [1].

**Table 7.** Usual notation of MAC-based service bundle authentication mechanism

<b>□ Usual Notation</b>	
<b>Message 1</b>	$SG \rightarrow OP : N_{SG}$
<b>Message 2</b>	$OP \rightarrow SG : N_{OP}, SJ, MAC_{OP}$
	$SJ : \text{Signed JAR}$
	$SS : \text{Shared Secret}$
	$MAC_{OP} = H(SJ, SS \parallel N_{OP} \parallel N_{SG})$

Table 8 shows an idealized protocol using BAN logic. (In this idealized form, we omit parts of the message that do not contribute to the beliefs of the recipient.):

**Table 8.** Idealized protocol of MAC-based service bundle authentication mechanism

□ BAN Idealization
Message 2 OP $\rightarrow$ SG : $N_{OP}, <H(SJ, SS N_{OP} N_{SG})>_{N_{SG}}$

We present assumptions to verify an idealized protocol as shown Table 9.

**Table 9.** Assumptions of MAC-based service bundle authentication mechanism

□ BAN Assumption
(1) $SG \models N_{SG}$ , (2) $SG \models \#(N_{SG})$ , (3) $SG \models SG \stackrel{N_{SG}}{\rightleftharpoons} OP$ ,
(4) $OP \models N_{OP}$ , (5) $OP \models \#(N_{OP})$ ,
(6) $SG \models SG \stackrel{SS}{\rightleftharpoons} OP$ , (7) $OP \models SG \stackrel{SS}{\rightleftharpoons} OP$

As mentioned above, we premise that only the embedded service bundle can access symmetric-key in the service gateway, and the operator is protected by a fire-wall. The final goal of service bundle authentication mechanism is to verify a service received from the operator. We present the goal using BAN logic in Table 10.

**Table 10.** Goal of MAC-based service bundle authentication mechanism

□ BAN Goal
$SG \models MAC_{OP}$ , $OP \models SG \models MAC_{OP}$
$SG \models OP \vdash SJ$

As a result, the service gateway should believe that the operator sends the service(the Signed JAR). To verify the goal, we use logical postulates in BAN logic.

The table 11 shows verification steps in message 2. The goal of service bundle authentication mechanism is achieved through step 1 and 2. That is, the service gateway can trust and use a service bundle sent by the operator.

**Table 11.** Verification of MAC-based service bundle authentication mechanism

<b>In Message2</b>
<b>STEP 1</b>
$SG \models SG \stackrel{N_{SG}}{\rightleftharpoons} OP, SG \triangleleft <N_{OP}, H(SJ, SS N_{OP} N_{SG})>_{N_{SG}}$
$SG \models OP \vdash (N_{OP}, H(SJ, SS N_{OP} N_{SG}))$
$SG \models OP \vdash H(SJ, SS N_{OP} N_{SG})$
<b>STEP 2</b>
$SG \models OP \vdash H(SJ, SS N_{OP} N_{SG}), SG \triangleleft (SJ, SS N_{OP} N_{SG})$
$SG \models OP \vdash (SJ, SS N_{OP} N_{SG})$
$SG \models OP \vdash SJ$

## 4 Conclusion and Future Work

We proposed the service bundle authentication mechanism in OSGi service platform, and verified it. From the verification result above, we achieved all of the authentication goals. That is, the results show that the service gateway and the operator trust each other in the key exchange process. Furthermore the results show that the service gateway can trust and use a service bundle sent by the operator.

In this paper, the efficiency and safety of the key exchange mechanism and the service bundle authentication mechanism is verified with the BAN logic. However we need to provide an experimental result to prove the efficiency of the mechanisms proposed in this paper. We also proposed a key exchange system for authenticating a service bundle and the service bundle authentication system, we further need to research and design a key exchange system for working together with user's authentication ticket.

## References

- [1] Young-Gab Kim, Chang-Joo Moon, Dae-Ha Park, Doo-Kwon Baik, "A MAC-based Service Bundle Authentication Mechanism in the OSGi Service Platform", DASFAA 2004 in *Lecture Notes in Computer Science*, p.137-145, Springer-Verlag, March 2004.
- [2] Young-Gab Kim, Chang-Joo Moon, Dae-Ha Park, Doo-Kwon Baik, "A Service Bundle Authentication Mechanism in the OSGi Service Platform", AINA 2004, p.420-425, IEEE Computer Society, March 2004.
- [3] John Clark, Jeremy Jacob, "A Survey of Authentication Protocol Literature: Version 1.0", University of York, Department of Computer Science, November 1997.
- [4] William Stallings, "Cryptography and Network Security", Pearson Education, 2002.
- [5] Marc Branchaud, "A Survey of Public Key Infrastructures", Department of Computer Science, McGill University, Montreal, 1997.
- [6] OSGi, "Secure Provisioning Data Transport using Http", RFC36, <http://www.osgi.org/>, 2002.
- [7] OSGi, "OSGi Service Gateway Specification - Release 3.0" <http://www.osgi.org>, 2003.
- [8] Michael Burrows, Martin Abadi, Roger Needham, "A Logic of Authentication", Digital Equipment Corporation, 1989.
- [9] Jan Wessels, Cmg Finance B.V. "Applications of BAN-Logic", 2001.
- [10] George Coluouris, et. al., "Distributed System", Edition 2, 1994.
- [11] A.D. Rubin, P. Honeyman, "Formal Methods for the Analysis of Authentication Protocols", CITI Technical Report 93-7, 1993.

# Optimal Data Dispatching Methods in Near-Line Tertiary Storage System\*

Baoliang Liu, Jianzhong Li, and Yanqiu Zhang

School of Computer Science and Technology, Harbin Institute of Technology, China  
{liubl,lijzh,sarai}@hit.edu.cn

**Abstract.** Many applications like digital libraries use Near-line Tertiary Storage Systems (TSS) to store their massive data. TSS consists of main memory, disks and tertiary devices. Typically highly referenced or recent data are stored on disks and historical data are stored on tertiary storage devices. We call it Data Dispatching: the determination of what kind of data should be stored on disks and what kind of data should be stored on tertiary storage devices. Traditionally it was up to the Database Management System (DBMS) administrator to dispatch data by hand. But DBMS has to take the responsibility if we want to bring tertiary storage devices under the control of DBMS. We proved in this paper that the data dispatching is an optimal problem and can be reduced to the famous binary knapsack problem. Experimental results showed that the average response time of TSS could be decreased by using optimal data dispatch method.

## 1 Introduction

Tertiary Storage Systems are required by many applications like digital libraries. In these systems, newly generated or highly referenced data are stored on disks and historical data or less referenced data are migrated periodically from disks onto tertiary storage devices. There are three schemes of using tertiary devices. In the first scheme, tertiary storage device is used as online device that data are all stored directly onto it and disk is used as buffer for tertiary resident data. Buffer replacement policies like LRU are used when disk buffer is full. Examples are Earth Observing Systems [10] and many other scientific applications. In the second scheme, tertiary storage device is used as offline device that is the archiving store. Data are stored on disks and DBMS doesn't know about the data on tertiary storage device. Data have to be loaded back onto disks by hand before they are processed. Examples are many applications used in Banks or Telecommunication companies. In the third scheme, tertiary storage device is used as near-line device on which data are first stored on disk and then migrated

---

\* Supported by National Natural Science Foundation of China under Grant No.60273082; the National High-Tech Research and Development Plan of China under Grant No.2001AA41541; the National Grand Fundamental Research 973 Program of China under Grant No.G1999032704.

onto tertiary devices periodically. Operation can be done on disks or on tertiary storage device directly. If there is remaining disk space, it is used as buffer for tertiary resident data. Examples are many video-on-demand (VOD) systems.

In the near-line tertiary storage system, how data are distributed on different devices in each storage level has much influence on the average system access time, since disks and tertiary storage device have different physical characters. Disks are fast and random access devices while tertiary storage device are slow devices. Intuitionally, highly referenced data should be stored on disk and less referenced data should be stored on tertiary storage device, but access frequency is not the only parameter we shall consider in data dispatching. Although near-line TSS have been used by many digital library systems, data dispatching problem hasn't been brought up previously to our knowledge.

The rest of the paper is organized as follows. In section 2 we discuss briefly the related works. In section 3, we prove that data dispatching problem can be reduced to the famous binary knapsack problem and optimal dispatching methods are presented. Experimental results in section 4 showed that by our data dispatching methods average system access time can be decreased. Finally we conclude this paper in section 5.

## 2 Related Work

The multi-level storage architecture was first brought up by Stonebraker in [4]. Postgres [5], Paradise [6] and TerraServer [11] are three example DBMS that can manage data on tertiary store. In these systems how data are dispatched to each storage level was not addressed.

Many works have been done on optimal data placement on tertiary storage device. The method of efficiently organizing large multidimensional arrays on tertiary storage devices were presented in [1, 2]. In [3], principles of optimal data placement in tertiary storage libraries are given. But all these works assumed that it was already known that what kind of data should be stored on tertiary storage devices, which was not always true.

The data pre-fetching methods from tertiary storage device onto disks were studied in [7, 8]. A Markov-Chain based prediction model was given in [7]. Several data pre-fetching method in VOD systems were given in [8], but no experimental results were presented.

## 3 Problem Definition

We assume that there are  $n$  chunks in TSS, which are represented by  $C = \{ \langle w_1, p_1 \rangle, \langle w_2, p_2 \rangle, \dots, \langle w_n, p_n \rangle \}$ .  $w_i$  represents chunk size and  $p_i$  represents chunk access possibility. Then the result of the data dispatching corresponds to a binary vector  $X = \langle x_1, x_2, \dots, x_n \rangle$ .  $x_i$  corresponds to the chunk  $i$  and  $x_i = 1$  if chunk  $i$  resides on disks and  $x_i = 0$  otherwise. Some parameters used in this paper are listed in Table 1.

To describe the problem clearly, we first give some definitions below:



**Table 1.** Parameters used in this paper

Parameters	Description
$Acc_D(i)$	Disk access time of chunk $i$
$Acc_T(i)$	Tertiary access time of chunk $i$
$pos(i)$	Positioning overhead of chunk $i$ when it's on tertiary device
$T$	Expecting access time of TSS
$S(D)$	Disk Income
$X = \{x_1, \dots, x_n\}$	Data Dispatching Vector
$X_D$	Data transfer rate of disk
$X_T$	Data transfer rate of tertiary storage device

**Definition 1 (Chunk Access Time).** *The time it takes to load chunk  $i$  into memory from disks or tertiary devices. It is denoted by  $Acc_D(i) = w_i/X_D$  if the chunk is stored on disks and  $Acc_T(i) = w_i/X_T + pos(i)$  if it is stored on tertiary devices.*

We assume that all disk accesses are multi-page I/O requests. The cost of a disk access is therefore derived by the amount of the data transferred. The disk seek cost and rotational latency are ignored. Tertiary data access time includes media switch time, positioning time of the tertiary storage device and data transfer time. We only consider positioning time and data transfer time. We ignore media switch time because through efficient I/O scheduling, the media switch time can be amortized into many tertiary requests.

**Definition 2 (System Access Time).** *The expecting time it takes to read a chunk into memory from disk or tertiary storage device in tertiary storage system. Denoted by:*

$$T = \sum_{i=1}^n x_i \cdot p_i \cdot Acc_D(i) + \sum_{i=1}^n \bar{x}_i \cdot p_i \cdot Acc_T(i) . \quad (1)$$

The goal of data dispatching is to find a binary vector  $X = \langle x_1, x_2, \dots, x_n \rangle$  to minimize the system access time.

**Definition 3 (Disk Income).** *Let  $Acc_D(i)$  be the access time if chunk  $i$  stored on disks and let  $Acc_T(i)$  be the access time if chunk  $i$  stored on tertiary storage device. Then we define Disk Income as:*

$$S(D) = \sum_{i=1}^N x_i \cdot p_i \cdot (Acc_T(i) - Acc_D(i)) . \quad (2)$$

Disk income represents the gains in access time if we store the chunks on disks not on tertiary storage devices. From (1) and (2) we can easily get:

$$T = \sum_{i=1}^N p_i \cdot Acc_T(i) - S(D) . \quad (3)$$

Since  $\sum_{i=1}^N p_i \cdot Acc_T(i)$  is a constant, so maximizing disk income is the necessary condition of minimizing system access time.

Now the problem of data dispatch reduced to maximize the disk income given disk size. This is the famous binary knapsack problem, in which the disk serves as the knapsack. There are two simple method to handle this problem, one is the exact method based on dynamic programming, the other is the approximate method based on greedy strategy. Readers are referred to [9] for details.

## 4 Experimental Evaluation

In this section we present simulation results about the system access time related to data dispatching method. The experiments have been carried out on the prototype system with simulated secondary and tertiary storage devices. The parameters of the devices used throughout all experiments are given in Table 2.

**Table 2.** Devices parameters used in the simulation experiments

Parameter	Secondary device	Tertiary device
Average Disk Seek Time	8ms	25ms
Average Rotational Latency	4.3ms	10ms
Disk Transfer Rate	11.5MB/s	3.4MB/s
Number of Volumes	-	8
Volume Exchange Time	-	10s

The dataset we used consists of 20000 chunks. The size of each chunk varies from 2MB to 10MB with average size of 5MB. And there are 61361 requests altogether.

In our simulation experiment, four different schemes were compared. These schemes are described below:

GD: Near-line architecture with greedy data dispatching method.

RD: Near-line architecture with random data dispatching method, which means the chunks are chosen to be stored on disk randomly.

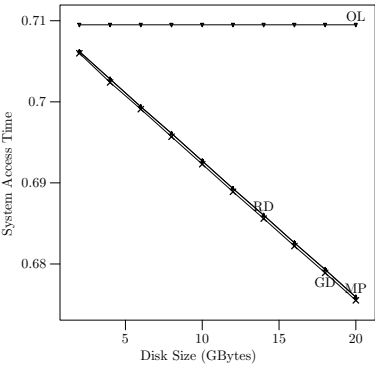
MP: Near-line architecture with max possibility data dispatching method, which means those chunks with maximum access possibility are stored on disk.

OL: Online architecture, which means all chunk are stored on tertiary devices.

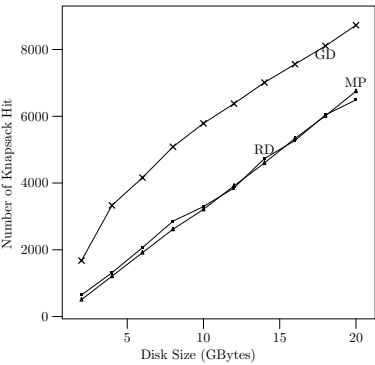
We didn't use the exact dynamic programming method because it will take too long time and too much memory space. In all these schemes, we chose LRU as buffer replacement strategy.

### 4.1 Performance Related to the Disk Size

In the first experiment, the disk size varied from 2 GBytes to 20 GBytes. The knapsack size was fixed half the disk size. The experiment results were listed in



**Fig. 1.** System access time related to size

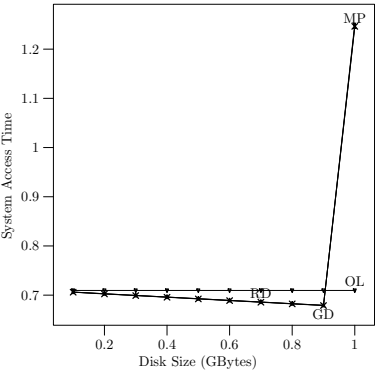


**Fig. 2.** Number of knapsack hit related to disk size

Fig. 1. The result showed that the system access time of all near-line schemes were better than that of online scheme. And the system access time of near-line schemes decreased as we increased the disk size. To see the result clearly, we listed the knapsack hit ratios in Fig. 2. The knapsack hit ratio was the number of requests met by the chunks in the knapsack to the total number of the requests. From the figure we could see that the knapsack hit ratio of greedy method was much larger than that of the other two near-line methods.

4.2 Performance Related to the Knapsack/Disk Ratio

In the second experiment, we fixed disk size at 2 GBytes. The knapsack/disk (K/D) ratio is varied from 0.1 to 1. When the K/D ratio equaled 1, it meant that all disk space was used as knapsack. We listed the result in Fig. 3. From the figure we can see that the system response time first decreased as we increased



**Fig. 3.** System access time related to knapsack/disk ratio

the knapsack size, and then there was a sharp increase of system access time when the K/D ratio was large than 0.9, the reason is that when all the disk space are used as knapsack there were no space for buffering tertiary resident data.

## 5 Conclusion

In this paper data dispatching methods for data on near-line tertiary storage systems are presented. We proved that data dispatching is a binary knapsack problem. Experimental results showed that greedy data dispatching method is the candidate because its performance is best and its overhead is very low.

We should follow the steps below when we design a near-line TSS: (1) Set aside enough disk space which would not be very large for buffering tertiary resident data. (2) Dispatch data onto disks and tertiary device using greedy data dispatching method. (3) Organize data on tertiary storage system using methods presented in [1–3].

## References

1. S.Sarawagi, M.Stonebraker.: Efficient Organization of Large Multidimensional Arrays. ICDE (1994) 328-336.
2. L.T.Chen, R.Drach, M.Keating, S.Louis, D.Rotem, A.Shoshani.: Efficient Organization and Access of Multi-Dimensional Datasets on Tertiary Storage Systems. Information Systems Journal **20** (1995) 155-183.
3. S.Christodoulakis, P.Triantafillou, F.A.Zioga.: Principles of Optimally Placing Data in Tertiary Storage Libraries. VLDB (1997) 236-245.
4. M.Stonebraker.: Managing Persistent Objects in a Multi-level Store. SIGMOD Conference (1991) 2-11.
5. S.Sarawagi.: Database Systems for Efficient Access to Tertiary Memory. In Proc. IEEE Symposium on Mass Storage Sys. (1995) 120-126.
6. J.Yu, D.DeWitt.: Query Pre-execution and Batching in Paradise: A Two-pronged Approach to the Efficient Processing of Queries on Tape-resident Data Sets. SS-DBM (1997) 64-78.
7. A.Kraiss, G.Weikum.: Vertical Data Migration in Large Near-Line Document Archives Based on Markov-Chain Predictions. VLDB (1997) 246-255.
8. P.Triantafillou, T.Papadakis.: On-Demand Data Elevation in a Hierarchical Multimedia Storage Server. VLDB (1997) 226-235.
9. Anany Levitin.: Introduction to The Design & Analysis of Algorithms. Addison-Wesley (2003).
10. J.Frew, J.Dozier.: Data Management for Earth System Science. SIGMOD Record-Volume **26** (1997) 27-31.
11. T. Barclay, D. R. Slutz, J. Gray.: TerraServer: A Spatial Data Warehouse. SIGMOD Conference (2000) 307-318.

# A Real-Time Information Gathering Agent Based on Ontology

Jianqing Li, Shengping Liu, ZuoQuan Lin, and Cen Wu

School of Mathematical Sciences, Peking University,  
Beijing, 100871 China  
{ljq, lsp, lz, wc}@is.pku.edu.cn

**Abstract.** This paper provides an agent based on an ontology that helps user to gather the real-time information. We present an approach of ontology representation based on concept graph and introduce the building and revising algorithms of knowledge base, then discuss the application of the ontology in information source representation. Experimental results show that the agent helps user accomplish the task effectively.

## 1 Introduction

At present people use search engine to search information from Web in general. It turns out some problems [1]. Firstly it needs a period to update indexing database, and can't be used for searching the real-time information, such as news information. Secondly current information-retrieval techniques which reflect only part of the document's content do not guarantee content matching [2].

Agent is a kind of software which assists people by accomplishing complicated task autonomously based on requirement [3] and includes a knowledge base and prover in general. Information agent is a kind of agent that retrievals information for user, whose knowledge base includes the facts of information source, web document and agent's ability etc. The prover controls agent's action and builds the strategy of searching. The core component of information agent is knowledge base, the efficiency of which limits agent's performance. There is a great need for ontological technique in information retrieval system based on agent for the reason that it improves the performance of knowledge base.

The reason of the low precision of IR systems is the difference between provider and consumer's ontology and incompleteness of information. Many works [2, 4, 5] show that ontology, which is used to describe document and user's query in semantic level, is a good way for solving these problems.

Ontology is the term used to refer to the share understanding of some domain of interest which may be used as a unifying framework to solve the problem [6]. Ontology necessarily entails or embodies some sort of world view with respect to a given domain. The world view is often conceived as a set of concepts, their definitions and their inter-relationships, which are referred to as a conceptualization. We use the concept graph to represent ontology in this paper.

The rest of the paper is organized as follows. Section 2 presents an approach of ontology representation based on concept graph. Section 3 introduces the building and revising algorithms of the knowledge base. Section 4 discusses ontological application. Section 5 describes the system architecture. Section 6 provides experimental results. Section 7 concludes the paper.

## 2 Ontology Representation

The semantic network is used to represent the complicated relations among objects and concepts in real world [7]. We adopt the idea from the semantic network and Sowa's concept graph [8] to represent ontology. A concept graph is a finite connected bipartite digraph which includes conceptual node and concept-associated node, and each concept-associated node connects with one or more conceptual nodes. All concepts compose a hierarchy. There is an inter-class relation between two concepts when a concept in upper level connects with a lower one, which is named as inter-class *IsKindOf* relation.

The element is interpreted as conceptual node of a concept graph in a domain's ontology-space, and the relationship among concepts is interpreted as concept-associated node. The level-relation among concepts is an inter-class *IsKindOf* relation. The hierarchy of concept graph is illustrated with the news domain in figure 1. Root is *News* class, which has subclass *Nation*, *Internation*, *Society* and *Sport* etc. Sports news has *Soccer*, *Basketball* and *Volleyball* etc. The relationship between *Nation* and *News* is an inter-class *IsKindOf* relation.

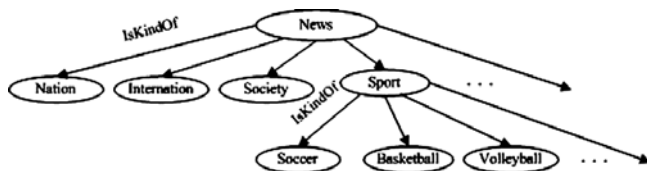


Fig. 1. A branch of ontology concept graph about news domain

We used a keyword's concept graph to define ontology, in which the concept term is in different representation-level with ontological concept graph. The keyword is regarded as a concept in keyword-graph. The relationships among the nodes in keyword-graph include the same semantic of Synonymy in WordNet [9].

## 3 Building and Revising the Ontological Knowledge

There are two components in the knowledge base: domain ontological space and ontological description. The process of building and revising the ontology space is performed by knowledge engineer through the interface agent's visual tool. The

building and revising the ontological description is done by machine learning. User also can build and revise the description through interface.

The building process takes a set of documents *Doc* in domain as input which include the ontological information, together with the domain's ontological set. The function parses the documents, and abstracts the keywords, then evaluates the word's effectiveness to characterize the ontology. The output is the ontological description which is a set of pairs of keyword and effectiveness value.

The revising process is a reinforcement learning process, which takes a set of the domain's documents *Doc* as input, which include the ontological information, together with the ontological set. The process parses the document, and abstract its words, then evaluate each word's effectiveness to characterize the ontology. If a word is included in description, the process revises its effectiveness value. If it is not, the process inserts the pair of word and effectiveness value into the ontological description.

## 4 Ontology Application

We will present an approach of information resource representation based on ontology in this section. The information resource is described in two levels in our system, which is coarsely granular and finely granular. The coarse description includes name, url and topic information. The fine description is to describe the content organization of a web site.

A research [10] shows that web site has a great deal of self-organization. The pages and links are created by users with particular interests, and same topical pages tend to cluster into natural "community" structures that exhibit an increased density of links, which exhibit the ontological knowledge about that topic which provides the foundation for provider to create the hyper links. Web pages are classified as hub and authority. A hub is a page that points to many authorities, whereas an authority is a page that is pointed to by many hubs. Characteristic patterns of hubs and authorities would be used to identify communities of same topical pages. The link analysis algorithm is designed for search the hub page and authorities page in a "community".

The agent describe a site's organization by hubs in this paper, which is defined as a quaternion  $HP := \langle URL, Title, Ontology, Site \rangle$ , where *Ontology* denotes a hub's topic, others are site and location's information. The ontological relation presents the topic relation among the hubs. The agent inferences a site's content organization through ontology and hub's quaternion. The information source knowledge base store the hub's quaternion of a site.

Agent acquires a site's hub through a topic-relate link analysis algorithm. The idea behind the link analysis algorithm is that the evaluation of a page is through hyper link. We use a improved PageRank [11] algorithm which has two phases. In the pre-processing phase, it filters pages which are independent with the topic or do not support the gathering constrain.

When information agent wants to gather some topic pages, it firstly searches the site's knowledge base for related hubs. If matched, agent begins gathering

page from these hubs and crawl the pages by broad-first policy. The agent will reach the interested community quickly following this policy and improve the search efficiency. Gathering agent can keep away from crawling area which is far from interested community under this strategy. The strategy accords with people's browsing behavior when he knows the site's organization.

## 5 Architecture

The agent solves the problems in two phases: gathering phases and querying phases, whose functional architecture is showed in Fig 2.

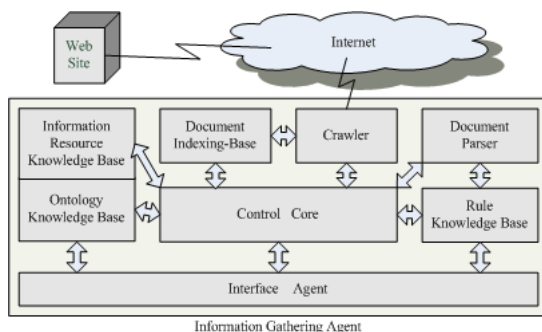


Fig. 2. Architecture of real-time information gathering agent

In the gathering phases the system accepts user's queries. Interface agent refines queries based on ontological knowledge base and user's selection, then submits the refined queries to system's control core which is composed of Scheduler, Planner and Executer. The scheduler builds a task description according to the queries and the information resource knowledge base, which includes task's goal and constrains. The planner gets the task and parses the task description as a sequence of subtask. The sequence will be distributed to executer through scheduler. The executer parses the sequence and assigns the subtask to correspondence solver. When the task is completed, the core returns the task's answer to user through interface agent.

In the querying phases the interface agent will enrich the query by the supporting of ontological knowledge base when system receives a query, then submits the query to the core. The core will search the local indexing base and retrieve the relevant documents and build a gathering task based on the query simultaneously. When the task completed, it puts the relevant documents to user.

## 6 Experimental Results

In order to perform the technical feasibility check, we built a prototype system, News Information Gathering Agent(NIGA). The NIGA successfully searched for



news page automatically from multiple news Web sites and the experimental results showed that it assists users to search and browse news page.

We use two parameters to evaluate NIGA, which are the depth average of corrected responses and precision of the gathered pages. The depth average of corrected pages is the average of corrected pages' gather-depth. The precision of the gathered pages is the percentage of correct responses out of the gathered answers. The former show the average cost of searching procedure in one aspect. The later show the efficiency of the gathering procedure. The performance is evaluated by comparing the output with the answer keys.

Fig. 3 shows the comparison of average searching depth of corrected pages for gathering news page in *news*, *sports news* and *science news* ontology from homepage and from hubs. The graph indicates that using knowledge of information resources reduces the average depth of collect pages, which means that the agent gets into the area which is the user interested community by link analysis function.

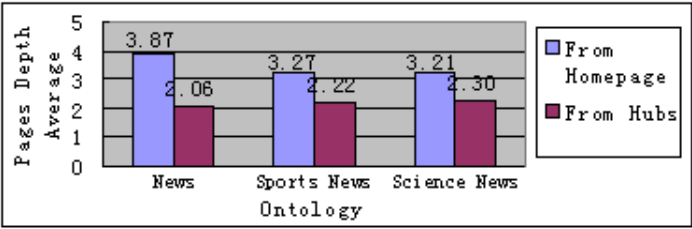


Fig. 3. Comparison of page's average depth from homepage and hubs

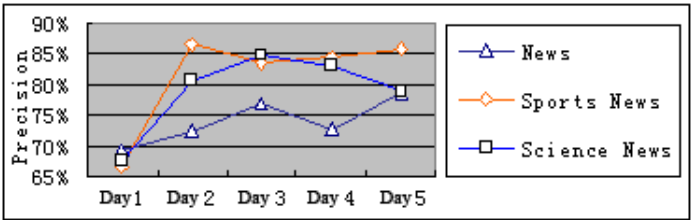


Fig. 4. The trendline of the precision to gather *news*, *sports news* and *science news* articles in five days

Fig. 4 shows the trendline of precision of the gathered pages in *news*, *sports news* and *science news* ontology, which is defined by equation below.

$$\text{Precision} = \frac{\# \text{ of relevant returned pages}}{\# \text{ of returned pages}} \quad (1)$$

The graph shows that the precision increases significantly from Day 1 to Day 2, and after Day 2 it decreases or increases slightly. This means that the agent is

sure to get into the area which is interested community after learning. Consequently the crawler will gather more relevant pages and more quickly than some other searching strategies.

The experimental results show that our agent based on ontology performs well to gather the real-time information from multiple Web sites.

## 7 Conclusion

Effective real-time information gathering on the Web is a complex task. Users have to exert a lot of efforts to browse and search the information they want. We present an agent that helps users to effectively gather the real-time information, which utilizes ontological knowledge. The experimental results show that our agent performs well to gather the real-time information from multiple Web sites.

The main features contributing to the effectiveness of a real-time information gathering agent are: the agent analyzes the link information of documents in sites and build resources representation based on ontology. The representation ensures that agents make the searching strategy based on the document distribution of resources, which improve the agent's performance. According to this work, the techniques based on agent and ontology resolve some problems of search engine and be applicable to gather real-time information from the Web.

## References

1. Peng Honghui and Lin Zuoquan. Search Engines and Meta Search Engines on Internet. *Computer Science*, 29(9):1-12, 2002.
2. Guarino, N.; Masolo, C. and Vetere, G. OntoSeek: Content-based access to the Web. *IEEE Intelligence Systems*, 14(3):70-80, 1999.
3. Woodridge, M. and Jennings, N.R. Intelligent Agent: theory and practice, *Knowledge Engineering Review*, 10(2):115-152, 1995.
4. Theilamann W. and Rothermel K.: Domain experts for information retrieval in the World Wide Web. In: Klush M, Wei G eds. *Lecture Notes in Artificial Intelligence*, Vol 1435. Berlin, Heidelberg, New York: Springer-Verlag, pp.216-227, 1998.
5. Wu Chenggang.: An Information Retrieval Server Based on Ontology and Multi-agent. *Journal of Computer Research & Development*, 38(6):641-647, 2001.
6. Borst, W.N. Construction of Engineering Ontologies. Ph thesis, University of Twente, Enschede, 1997.
7. Lin Zuoquan and Shi Chunyi. Formalization of Inheritance Reasoning in Semantic Networks. *Pattern Recognition and Artificial Intelligence*, 4(2):33-43, 1991.
8. Sowa, J.F. Conceptual structure: Information processing in mind and machine. Addison-Wesley Publisher, Reading, Mass. 1984.
9. Miller, G.A. WordNET: A lexical database for English. *Communications of the ACM*, 38(11):39-41, 1995.
10. Kleinberg, J. The Structure of the Web. *Science*, 294(9):1849-1850, 2001.
11. Brin, S. and Page, L. The anatomy of a large-scale hypertextual Web search engine. In 7th International World Wide Web Conference, Brisbane, Australia, April, 1998.

# Dynamical Schedule Algorithms Based on Markov Model in Tertiary Storage\*

Yanqiu Zhang, Jianzhong Li, Zhaogong Zhang, and Baoliang Liu

Department of Computer Science of Technology, HIT, Harbin, China  
sarai@263.net

**Abstract.** Tertiary storages, such as tape libraries and optical disc libraries, are becoming the important storage devices in massive data applications. With the massive storage space they have, the tertiary storages have very low access efficiency. Designing good schedule algorithms is an important method to improve the access efficiency in tertiary storage. Stochastic Markov model is used for predicting the expected number of accesses to the data on tertiary storage. Two new schedule algorithms named MarkovMR and MarkovSRF are given also. Generally, in a tape library, there is a robot arm, a few of tape drives and a magazine where a lot of tapes located in. When the tapes are kept in tape drives, we call them online tapes. Otherwise, we call the tapes in the magazine offline tapes. Weight factors are used to above schedule algorithms to favor online tapes so that the requests for online tapes are served first before online tapes are ejected. The weighted algorithms are named wMarkovMR and wMarkovSRF. By compared to the Round-robin policy, the experimental results show that the four schedule algorithms based on Markov model have higher efficiency of data access in tertiary storage. The efficiency of wMarkovMR is highest among all the algorithms. Furthermore, the optimal factors can be derived from experiments.

## 1 Introduction

With the development of information technique, there is a very large scale of data in the world. For example, there are about  $10^{15}$  bytes returned from the observing satellites in NASA each year. In recent years, there are  $10^{12}$  bytes of observing data in all kinds of remote sense systems of China each day. Obviously, the traditional secondary storage system (primary memory and magnetic disk) has limitation to accommodate so much massive data. The tertiary storage (primary memory, magnetic disk and the tertiary devices) is becoming more important than before.

Tape library and optical disc library are the main tertiary storage devices. With the massive storage space they have, the access performance is lower 3-4 magnitude than that of magnetic disk because of the mechanism operations in tertiary storage. Both the long tape switch time and very few robot arms require good schedule algorithms

---

\* Supported by the National Natural Science Foundation of China under Grant No.60273082; the National High-Tech Research and Development Plan of China under Grant No.2001AA41541; the National Grand Fundamental Research 973 Program of China under Grant No.G1999032704.

to decrease the times of tape switch and improve the utilization rate of tape drives. In this paper, stochastic Markov model is used for predicting the expected number of accesses to the data on tapes. We propose two new schedule algorithms, which are MarkovMR and MarkovSRF, based on Markov model in tape library. Furthermore, we use weight factors in above schedule algorithms to favor the online tapes so that the requests for online tapes are served first before the online tapes are ejected. The weight schedule algorithms are named wMarkovMR and wMarkovSRF respectively which keep tapes with high access probability online as possible as they can. Compared to the Round-robin policy, the experimental results show that the four schedule algorithms based on Markov model have higher efficiency of data access in tertiary storage.

## 2 Related Works

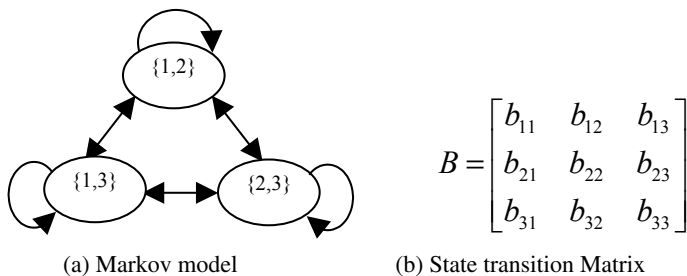
In order to improve the efficiency of data access in tertiary storage, optimal data placement policy based on tertiary storage [1-4], data management and index techniques in tertiary storage [5-7] and data I/O schedule algorithms are researched in recent years. For single tape I/O schedule algorithms, there are READ algorithm, FIFO algorithm, OPT algorithm, SORT algorithm, SLTF algorithm, SCAN algorithm, WEAVE algorithm, LOSS algorithm, MScan algorithm and MScan\* algorithm [8,9]. The multi-tape I/O schedule algorithms have to consider the characteristic of the tape library with more than one tape drives and a lot of tapes. The multi-tape I/O algorithms consist of two steps which are tape selection and single tape I/O schedule [10-14].

## 3 Schedule Algorithms in Tertiary Storage

### 3.1 Problem Definitions

In old schedule algorithms, only one tape is selected according to tape access probability. We select all tapes for tape drives in our algorithm. In order to describe the algorithm clearly, we give some definitions as follows. The robot locate time, named  $T_{\text{robot}}$ , is the whole time of the process of unloading the online tape from the tape drive, of laying the tape back to the magazine, and of fetching the offline tape into the tape drive. By coding the tapes in a tape library, there is a TapeID for each tape. If a tape drive contains an online tape whose TapeID is  $i$ , the tape drive state is  $i$ . If the tape drive is idle, the tape drive state is 0. All the tape drive states compose a state set named a tape library state.

For an example of the Markov model in figure 1(a) with 2 tape drives and  $T_s=3$  tapes in a tape library. In figure 1 (a), the three circles describe three tape library states, and the numbers in circles are TapeIDs of online tapes. If some tapes are needed to exchange, the state transition happens between two tape library states.



**Fig. 1.** The Markov model and State transition Matrix with 2 tape drives and 3 tapes

Let  $b_{ij}$  denotes the transition probability from tape library state  $s_i$  to  $s_j$ . All the transition probabilities  $b_{ij}$  form a matrix  $B$  called tape library state transition probability matrix as in figure 1(b). Let  $ET = \{s_j - s_i\}$  represents the TapeIDs should be loaded during state transition from  $s_i$  to  $s_j$ , then  $b_{ij} = \sum_{k \in ET} p^k$ . Here,  $p^k$  is the  $k^{\text{th}}$  tape access probability in current request waiting queue.

If current tape library state is  $s_i$ , for each  $s_j, j = 1, \dots, n$ , the tape switching cost  $c_{ij}$  produced by the state transition from  $s_i$  to  $s_j$ , then the corresponding state producing probability is  $PROB_j = f(b_{ij}, c_{ij}), j = 1, \dots, n$ . Here,  $f(b_{ij}, c_{ij})$  is a function of  $b_{ij}$  and  $c_{ij}$ . The next optional tape library state is  $s_{op} = \{s_k \mid \max_{k=1, \dots, n} PROB_k\}$  or  $s_{op} = \{s_k \mid \min_{k=1, \dots, n} PROB_k\}$ . We propose a notation named request window used to select a group of requests in request waiting queue.

### 3.2 The Markov-Based Schedule Algorithm with Max Requests

In Markov-based schedule algorithm with max requests, abbreviated with MarkovMR, for each tape library state, we compute the total number of requests of all tapes included by this tape library state. Then the tape library state with max number of requests is selected. The description of the MarkovMR is following:

- 1) Applied request window on request waiting queue,  $|W|$  requests are selected. We compute each tape library state in request window. We select  $s_{now} = \max_{i=1, \dots, n} s_i$  is the first tape library state.
- 2) We delete the requests which are included in online tapes from the request waiting queue. These requests are inserted into the robot waiting queue. The robot arm will fetch the tapes with TapeIDs in  $s_{now}$  into the tape drives.
- 3) If the request waiting queue is NULL, go to step 8). The algorithm is finished. Else, go to step 4).

- 4) We compute the transition probability matrix  $B$  according to the requests in current request window.
- 5) According to the transition probability matrix  $B$ , we compute the state producing probabilities of each tape library state.

$$PROB_j = f(b_{ij}, c_{ij}) = \sum_{k \in s_j} p^k - c_{ij}, j = 1, \dots, n.$$

$$\text{Here, } p^k = \sum_{q \in W} f(q), f(q) = \begin{cases} 1, & \text{if } q \in W \text{ and } q \in k^{\text{th}} \text{ tape} \\ 0, & \text{else} \end{cases}.$$

- 6) We select the next tape library state is  $s_{op} = \{s_k \mid \max_{k=1, \dots, n} PROB_k\}$ . Let  $ET = \{s_{s_{op}} - s_{s_{now}}\}$  which includes TapeIDs needed to be loaded into tape drives.
- 7) Let  $s_{now} = s_{op}$ ; go to step 3).
- 8) The algorithm is finished.

### 3.3 The Markov-Based Schedule Algorithm with Small Request First

In Markov-based schedule algorithm with small request first, abbreviated with MarkovSRF, for each tape library state, we compute the total request size of all tapes included by this tape library state. Then the tape library state with smallest request size is selected. In MarkovSRF algorithm, the  $k^{\text{th}}$  tape access probability is computed

$$\text{as } p^k = \sum_{q \in W} f(q), f(q) = \begin{cases} \text{sizeof}(q), & \text{if } q \in W \text{ and } q \in k^{\text{th}} \text{ tape} \\ 0, & \text{else} \end{cases}, \text{ and the}$$

$$\text{state producing probability is } PROB_j = f(b_{ij}, c_{ij}) = \sum_{k \in s_j} p^k + c_{ij}, j = 1, \dots, n.$$

Then the next optional tape library state is  $s_{op} = \{s_k \mid \min_{k=1, \dots, n} PROB_k\}$ .

### 3.4 The Weighted Markov-Based Schedule Algorithm with Max Requests

In order to decrease the cost of tape switch furthermore in MarkovMR algorithm, we use a weight  $\alpha$  to favor the online tape so that more requests are served before the online tapes are rejected. The algorithm of MarkovMR with a weight is named

wMarkovMR. A vector  $F = \{f_i, i = 1, \dots, T\}$ ,  $\hat{f}i = \begin{cases} \alpha, & \text{if } i \text{ is the online } \textit{tapeID} \\ 1, & \text{if } i \text{ is the offline } \textit{tapeID} \end{cases}$

is used in wMarkovMR. The difference between MarkovMR and wMarkovMR is the computation of state producing probability in above description of algorithm. Here,

$$PROB_j = f(b_{ij}, c_{ij}) = \sum_{k \in s_j} p^k * f_k - c_{ij}, j = 1, \dots, n.$$

3.5 The Weighted Markov-Based Schedule Algorithm with Small Request First

The algorithm of MarkovSRF with a weight  $\beta$  is called wMarkovSRF. A vector

$$F = \{f_i, i = 1, \dots, T\}, \hat{f}_i = \begin{cases} \beta, & \text{if } i \text{ is the online } tapeID \\ 1, & \text{if } i \text{ is the offline } tapeID \end{cases}$$
 is used in wMarkovSRF.

The difference between MarkovSRF and wMarkovSRF is the computation of the state producing probability in above description of algorithm. Here,

$$PROB_j = f(b_{ij}, c_{ij}) = \sum_{k \in s_j} p^k * f_k + c_{ij}, j = 1, \dots, n.$$

4 Experimental Evaluations

The experiments have been carried out on the prototype system with simulated tertiary storage devices. Here, we used the Ampex tape library with 2 tape drives and 20 tapes as the tertiary storage device [15]. We write the schedule algorithms in C++ language in Microsoft window 2000 professional operation system. We simulate the requests arriving at uniform random distribution. The number of requests varies from 100 to 1000. We select the SORT algorithm [10] as the single tape I/O schedule. The four schedule algorithms in this paper are used to select tapes into tape drives.

First, we compare the response time of five different schedule algorithms with 2 tape drives and 4 tapes in Ampex tape library in Figure 2. We can see that the response times of our four algorithms are all lower than that of Round Robin algorithm.

The impact of weight in wMarkovMR is shown in figure 3. Here, the number of requests is 1000, with 10 tapes and 2 tape drives. We can see that with the weight  $\alpha$  increases, the response time decreases. When  $\alpha$  increase from 1.2 to 1.9, the response time decreases dramatically. When  $\alpha$  is 1.9, the response time arrives the lowest point. When the value of  $\alpha$  is over 2, the response time of wMarkovMR keeps constant value. It means the value of  $\alpha$  is convergent. The same situation can be seen in wMarkovMR. However, with the value of  $\beta$  decreases, the response time of wMarkovSRF decreases too. The experiments show that the weights in schedule algorithms improve the access efficiency in tape library.

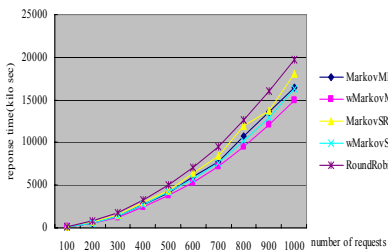


Fig. 2. Comparison of response time

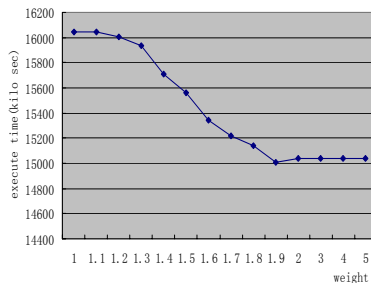


Fig. 3. Weight in wMarkovMR

## 5 Conclusions

Tape libraries and optical disc libraries are the important tertiary storage devices. With the massive storage space they have, the tertiary storages have very low access efficiency. In order to improve the access efficiency in tertiary storage, four new schedule algorithms based on Markov model are proposed in this paper which are MarkovMR, MarkovSRF, wMarkovMR and wMarkovSRF. Compared to the Round-robin policy, the experimental results show that the schedule algorithms based on Markov model have higher access efficiency in tertiary storage. Especially, the MarkovMR and the wMarkovMR appear to be the better schedule algorithms which suit tertiary storage system.

## References

1. S. Sarawagi and M. Stonebraker. Efficient organization of large multidimensional arrays. In Proceedings of the Tenth International Conference on Data Engineering, Houston, Texas, USA., pages 328--336. IEEE Computer Society, 1994. pp. 328-336.
2. Ford DA, Christodoulakis. Optimal Placement of High-Probability Randomly Retrieved Blocks on CLV Optical Discs. ACM Transactons On Information System, 1991, 9(1):1-30.
3. Triantafillou P, Christodoulakis S, Georgiadis C. Optimal Data Placement on Discs: A Comprehensive Solution for Different Technologies. HERMES Technical Report. Multimedia Systems Institute of Crete, Greece. (1996)
4. Christodoulakis S, Triantafillou P, Zioga F. Principles of Optimally Placing Data in Tertiary Storage Libraries. In: VLDB Conf., 1997, Athens, Greece, pp236-245.
5. A. Shoshani, L. M. Bernardo, H. Nordberg, D. Rotem, and A. Sim, Multidimensional Indexing and Query Coordination for Tertiary Storage Management, (SSDBM '99).
6. A. Shoshani, A. Sim, L. M. Bernardo, H. Nordberg. Coordinating Simultaneous Caching of File Bundles from Tertiary Storage. 12th International Conference on Scientific and Statistical Database Management (SSDBM'00).
7. Theodore Johnson: Coarse Indices for a Tape-Based Data Warehouse. Proceedings of the Fourteenth International conference on Data Engineering, Orlando, Florida, USA. IEEE Computer Society, 1998:231-240.
8. A. K. Hillyer and A. Silberschatz. Random I/O scheduling in online tertiary storage. In Proc. ACM SIGMOD Int. Conf. on Management of Data, Canada, 1996. 195 – 204. (Other references are omitted).



# Image Retrieval Using Structured Logical Shape Feature

Nanhyo Bang and Kyhyun Um

Dept. of Computer and Multimedia Engineering, Dongguk University,  
3 Ga 26 Pil-dong, Jung-gu, Seoul, 100-715, Korea  
{jjjang, khum}@dgu.ac.kr

**Abstract.** Most visual information retrieval systems on the web treat image as an object that is annotated with some keywords. However many semantic contents that cannot be expressed with some keywords are included in one image. It is the reason why most commercial and research systems utilize shape as the most cognitive information to represent contents of an image and to differentiate an image from others. In this paper, we propose an extraction method of logical shape feature to reflect structure of shape and coarse-fine matching method using it. For similar retrieval, we generate pattern segment matrix that is composed of curves's type in order to find the most similar curve sequence. We use it for coarse-fine matching because our shape features have global characteristic as a structural feature and local characteristic as an adaptive feature of shape. A pattern segment matrix has the advantage to search with only a small quantity of structural features. Our experiments show that structural-adaptive features through logical analysis result in effectively classifying shapes according to their cognitive characteristics. Various experiments show that our approach reduces computational complexity and retrieval cost.

## 1 Introduction

Nowadays, the web is regarded as a huge data repository for many information retrieval systems that use images. In such systems, the importance of management and search method of image grows much larger [1]. Various researches on the web data have concentrated mainly on a text-oriented web document. Most visual information retrieval systems on the web treat image as an object that is annotated with some keywords. However a lot of semantic contents that cannot be expressed as some keywords are included in one image. Researchers who have studied content-based retrieval systems classify features that compose image into a primitive feature and a logical feature [2]. A logical feature means a complex content, including semantic information. In [3], content is defined as “an understanding of the semantics of objects”. In this aspect, the shape feature among many features that represent image has been widely used by virtue of its ability to recognize similar images. However, most researches using shapes have represented shape as primitive feature, and this brought results that person cannot understand.

Therefore, we propose image retrieval method using logical shape feature. For this, we logically analyze shape information and to extract structural features and adaptive features. We used a curve segment as a fundamental element to express the appearance of a shape. Logical contents of shape are represented using curve segments with

adaptive features and structural relation between the units. Structural and adaptive features are extracted through a logical and structural analysis using a curve segment. The structural feature as a global feature is used as a coarse-matching feature because it contains pattern and relationship information of curve segments comprising the whole shape. The adaptive feature, on the other hand, is a local feature. Using this feature produces correct results because of its ability to reflect specific geometrical properties of each pattern segment. We generate pattern segment matrix in order to find the most similar curve segment sequence. A similar retrieval to find shapes that have most similar structure is easily executed because this matrix is composed of only type of pattern segment. Our experiments show how retrieval cost is reduced when conducting a search using these structural- adaptive features.

This paper is organized as follows. In chapter 2, we survey related works. In chapter 3, we explain a logical analysis scheme of the shape and structural and adaptive features. A matching process and similarity function is explained in chapter 4. Experiment results and performance analysis are shown in chapter 5. A conclusion is drawn in chapter 6.

## 2 Related Works

Many researches use contour points of an object because they contain most intuitive information of shape. The main topics covered by contour-based researches include matching strategies with contour points or additional information such as curvature of shape as well as methods for interest point extraction that preserves the original shape contour while reducing dimensionality [4][5]. However, poor search performance is a common problem among researches that use these primitive features. Multi-dimensional, variable shape features bring out serious performance degradation on matching and similarity search [6]. As an effort to solve these problems, concave/convex segment is extracted from the contour as shown in [7], and similarity search is conducted using dynamic programming. Among others, a curve-based representation and matching method of shape has been widely used because it has the advantage of computational efficiency. However, these methods don't represent interior of the shape and spatial relationship. It cannot adequately distinguish between perceptually distinct shapes whose outlines have similar features [9].

## 3 Structural Analysis Scheme for Shape Feature

We define a shape as a set of curve segments with a pattern. At preprocessing step, we extract the contour points from image's object to represent shape. However, because the size of contour points is too large, we use an adequate number of interest points that reflect a characteristic of shape. Interest points are aligned clockwise and described as a set  $S = \{p_1, p_2, \dots, p_n\}$ , where  $p_i$  denote an interest point and subscript  $n$  is the index number of an aligned interest point. A  $p_c$  is the gravity point of a shape. Two basic features are extracted from these interest points:  $\theta(\angle p_1 p_c p_k)$  and

$\alpha$  ( $\angle p_{i-1}p_i p_{i+1}$ ) feature. In our work, a curve is separated using  $\alpha$  feature of interest points that comprises a concave part. The separated segment is defined as curve segment,  $CS = \{p_i \mid is \leq i \leq ie, f\angle A(p_{is}), f\angle A(p_{ie}) > 180^\circ\}$ . If a CS includes two interest points, it becomes a concave segment. A CS merges with the next, if the next CS is also a concave segment. The CSs have specific patterns according to logical property: convex/concave property, turning property. Logical properties are characteristics that logically define the curve pattern of CS. We redefine CS with logical properties as PS(Pattern Segment), and we generate PSS(Pattern Segment Sequence). The PSS is defined as  $PSS = \{PS_i \mid PS_i = \langle pt, sflist, aflist \rangle, 1 \leq i \leq s\}$  where  $pt$  is a pattern type,  $sflist$  is structural features, and  $aflist$  represents adaptive features of PS. The logical properties of CS are expressed as adaptive features.

When the pattern types of PSS of two shapes are the same one, their structural features are used to calculate the structural difference of the shapes. These features are as follows : a)  $\alpha$  feature of a PS's last interest point,  $\alpha dist$ , b) Pattern range,  $pr$  - The start base angle ( $\angle p_{is+1}p_{is}p_{ie}$ ) and end base angle ( $\angle p_{ie-1}p_{ie}p_{is}$ ) features are appended in order to solve the problem that  $\alpha$  features of two shapes are the same but appear to be different, c) Size ratio of PS,  $sr$  - we extract the proportion of a PS to whole shape.

Adaptive features are features that reflect the logical and geometrical properties of each PS. Representative adaptive features are as follows: a) Average of  $\alpha$  feature,  $cc\alpha$  - This feature represents the degree of circularity, b) Eccentricity  $e$  - This represents the eccentricity of the polygonal segment, c) Turning angle  $ta$  - A  $ta$  is turning angle value of PS with partial or global turning properties.

## 4 Matching

Similarity retrieval is executed using PSS. First, a shape of query is classified into specific type of shape. Then we filtered shapes that are most similar according to the structural information of PSS in order to reduce search cost.

For structural matching, the PSS feature of shape D in the database is compared with one of query shape Q. The simple classification is carried out at first step. A shape of query is classified following polygon: convex polygon(s of QPSS=0), simple curved polygon(s of QPSS=1), turning polygon, complex polygon(s of QPSS  $\geq 2$ ). Following matching steps are for a complex polygon. A Pattern Segment Matrix (PSM) is generated in order to find a comparably similar segment sequence. PSM expresses a value to represent the identity of pattern types between the PSSs of two shapes as elements of the matrix:

$$M_{rc} \text{'s value} = \begin{cases} 1, & QPS_r.pt = DPS_c.pt \\ 0, & QPS_r.pt \neq DPS_c.pt \end{cases} \quad (QPS_r \in QPSS, DPS_c \in DPSS)$$

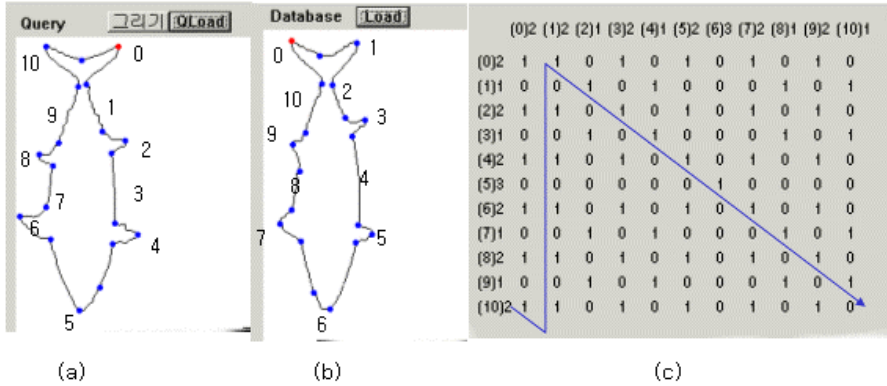
PSS of Q (QPSS) and PSS of D (DPSS) are used to represent the row and column axis of the matrix, M. After generating the matrix, we test whether the value of 1 continuously appears in a clockwise direction if M's value is 1. Again, we test the PSM made with QPSS, which was generated in reverse to find the symmetric shape. CSP(Candidate Similar Pattern sequence) are sequential segments with a similar pat-

tern type between Q and D. CSP represent PSs with similar pattern in a diagonal direction in PSM. The CSP is as  $CSP = \{CSP_i | CSP_i = \langle sq, sd, sleng \rangle\}$  where,  $sq$  stands for index  $r$  of QPSS, the starting point of sequence, and  $sd$  is  $c$  index of DPSS, and  $sleng$  is length of the sequence.

The  $rCSP$ , the segment sequence most similar to the shape of query and shape of database, is selected from PSM.  $rCSP$  is a CSP that is selected due to structural matching.

$$rCSP = MAX((\alpha dist_i + pr_i + sr_i)/3), (\alpha dist_i, pr_i, sr_i \in \forall CSP_i) \quad (1)$$

Among CSPs, we select  $rCSP$  with a maximum similarity value between structural features. In Fig.1, a point on the contour of shape expresses interest point. A number on the curve is aligned index of pattern segment. The (c) of Fig.1 is a sample PSM that is made by shapes of (a)(b). A line on PSM denotes  $rCSP$  that length is 11.



**Fig. 1.** (a) sample shape of query, (b) sample shape of database, (c) sample PSM

For fine matching, the adaptive features of two PSs included in  $rCSP$  are compared.  $Dist(aflist)$  expresses the similarity between adaptive features of two PSs. The total similarity cost is defined as

$$SimCost = Min(\frac{Qr + Dr}{2}, \frac{2Dr}{Qr + Dr}) \times Dist(aflist) \quad (2)$$

The  $Qr$  and  $Dr$  are the ratio of PSs belonging to  $rCSP$  in the whole shape. If a value of  $Dist(aflist)$  is very high (similar), but difference of  $Qr$  and  $Dr$  is large then  $SimCost$  reduce similar ratio of two shapes below arithmetic mean value.

## 5 Experiments

Our experiments show that similar shapes are quickly filtered with only structural and adaptive features. We use two test databases : 1100 shapes from SQUID and 216

shapes selected from MPEG-7 test database. On an average, the extracted features are as follows: For SQUID, contour point is 693, interest points are 22 and PS is 8. For the second database, contour point is 878, interest points are 23 and PS is 7.

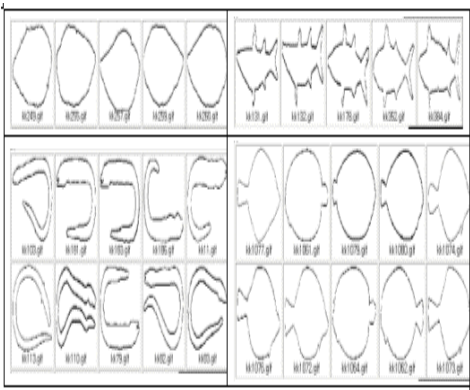


Fig. 2. Classified shapes

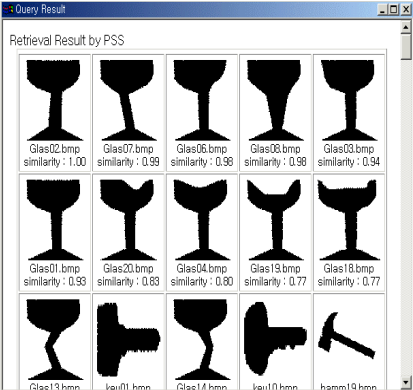


Fig. 3. Query result

Fig. 2 shows classified shapes using only the PSS and logical properties from SQUID data. Fig.3 is a result from the second test database, which has similar candidate shapes of partially various transformations that are high similarity.

We do a performance evaluation of shape features with features that are used such as storage cost and feature extraction cost. We compare our scheme with some representative methods such as Grid Based and the Fourier Descriptors Method in [8].

Table 1. Storage cost and feature extraction cost comparison

Criteria	Grid Based Method	Fourier Descriptors Method	ours
Storage cost(byte)	$2*((mj/gn)^2)/8+8$	$8 * fn$	$13 * m$
Extraction cost	To find major axis $O(N^2)$ To find minor axis $O(N)$ To rotate and scale $O(N)$ To generate a binary number $O(N^3)$	To find centroid $O(NlogN)$ To find all radii and compute FD coefficients $O(r^2)$ To generate signature $O(r)$	To Extract interest point $O(NlogN)$ To segment PSS $O(m)$ To extract structural-adaptive features $O(m)$

In Table 1, mj is the major axis size and gn is the grid cell size for the grid-based method. fn is the number of Fourier coefficients and r is the number of radii for the Fourier Descriptors Method. In our scheme, m is the number of interest points. Our scheme shows better performance from the point of view of feature extraction cost. This is almost the same as the generation time of TPVAS in [8]. At our scheme, computation cost to make pattern matrix is  $O(s^2)$  and one to make CSPs is  $O(s^2)$ . A computation cost to generate rCSP is  $O(cs)$ . A s is the number of a curve, and cs is the

number of CSP. Our scheme has a merit to search with only a small quantity of structural features because  $s$  is so small number that it is about 1/3 of interest points.. A retrieval performance is improved by changing average similarity ratio of *SimCost* that is used arithmetic mean into  $Min(\frac{Qr + Dr}{2}, \frac{2Dr}{Qr + Dr})$ .

## 6 Conclusion

In this paper, we suggest image retrieval method using logical shape feature. A shape feature is extracted from logically analyzed structural scheme of shape. After simple classification is performed about query shape, structural matching is executed about complex polygonal shapes through pattern segment matrix. Our scheme using matrix is efficient because we use structural feature of small size, not all shape features. Since PS reflects local features, it can be used for searching partial or overlapped objects. Experimental results show that our shape feature adequately represents object shape. The designed logical properties are applied to various shapes of closed polygons. These logical features extend meaningful semantic features by integrating with other features. The experiments show that our approach is close to human visual perception.

## References

- [1] Mark E.Hoffman, Edward K., Wong, "Content-based image retrieval by scale-space object boundary shape representation", Storage and Retrieval for Media Databases, Proc. of SPIE 2000, Vol.3972, pp.86-97
- [2] Venkat N.Gudivada, Vijay V.Raghavan, "Content-Based Image Retrieval Systems", IEEE Computer, Sep., 1995, pp.18-22
- [3] A.Desai Narasimhalu,"Special section on content-based retrieval", ACM Multimedia Systems, 1995, No.3, pp.1-2
- [4] Corney, J., Rea, H., Clark, D., Pritchard, J., Breaks, M.; Macleod, R. "Coarse filters for shape matching", IEEE Computer Graphics and Applications, Vol. 22 Issue: 3, May/June 2002, pp. 65-74
- [5] Dengsheng Zhang; Guojun Lu., "Generic Fourier descriptor for shape-based image retrieval", IEEE Intl. Conf. on Multimedia and Expo 2002. vol. 1, pp. 425-428
- [6] Suganthan, P.N., "Shape indexing using self-organizing maps", IEEE Transactions on Neural Networks, vol. 13, Issue: 4, Jul 2002, pp. 835-840
- [7] Mignotte, M., "A new and simple shape descriptor based on a non-parametric multiscale model", Proc. of Intl. Conference on Image Processing, Vol.: 1, 2002, pp. 445-448
- [8] Maytham Safar, Cyrus Shahabi and Xiaoming Sun, "Image Retrieval By Shape: A Comparative Study," IEEE Intl. Conference on Multimedia and Expo (I),2000, pp.141-154
- [9] Thomas B. Sebastian, Philip N. Klein, Benjamin B. Kimia, "Alignment-based Recognition of Shape Outlines", Lecture Notes in Computer Science(2001) .vol.2059, pp.606-617

# Automatic HTML to XML Conversion

Shijun Li<sup>1</sup>, Mengchi Liu<sup>2</sup>, Tok Wang Ling<sup>3</sup>, and Zhiyong Peng<sup>4</sup>

<sup>1</sup> School of Computer, Wuhan University, Wuhan, China 430072

shjli@public.wh.hb.cn

<sup>2</sup> School of Computer Science, Carleton University,  
1125 Colonel By Drive, Ottawa, ON, Canada K1S 5B6

mengchi@scs.carleton.ca

<sup>3</sup> School of Computer, National University of Singapore,  
Lower Kent Ridge Road, Singapore 119260

lingtw@comp.nus.edu.sg

<sup>4</sup> State Key Lab of Software Engineering, Wuhan University,  
Wuhan, China 430072

peng@whu.edu.cn

**Abstract.** We present a new approach to automatically convert HTML documents into XML documents. It first captures the inter-blocks nested structure, then the intra-blocks nested structure, which consists of blocks including headings, lists, paragraphs and tables in HTML documents, by exploiting both formatting information and structural information implied by HTML tags.

## 1 Introduction

As most web documents are in HTML, automatic understanding of HTML documents has many potential applications. How to capture the nested structure in HTML documents and convert it into XML is a significant challenge.

The main part of an HTML document is the *body* element, which consist of a sequence of *blocks* that may be *heading*, *paragraph* (*p* element), *list*, and *table*. A sequence of *blocks* partitioned according to HTML grammar may not be in the same semantic hierarchy. Thus, the first important step is to capture the inter-block nested structure in an HTML document. As a *heading* element, briefly describes the topic and nested structure, we can capture the inter-block nested structure of an HTML document according to different levels of *heading* in an HTML document. However, there are some HTML documents, in which the authors use formatting information rather than *heading* to show the nested structure. So we must exploit the formatting information that includes *bold*, *strong* tags and the biggest font etc. to capture the inter-block nested structures.

After capturing the inter-block nested structure, the next step is to capture the intra-block nested structure that includes *paragraph*, *list* and *table*. We exploit both structural information that HTML tags imply, such as a *list* consists of *items*, and formatting information to capture the intra-block nested structure.

HTML tables may have nested headings. To capture their nested structure, we need normalize HTML tables by inserting redundant cells into them. Based

on the normalized table, we can map the attribute-value pairs to XML documents. To convert HTML tables without marked headings via *th* element, the key is recognizing their headings. We introduce the notion of *eigenvalue* in the formatting information and a heuristic rule to recognize the headings of HTML tables. Without losing generality, we only discuss the well-formed HTML document. Since for those HTML documents that are not well-formed, we can use a free utility HTML TIDY by W3C to convert them into well-formed HTML ones.

This paper is structured as follows. Section 2 introduces the approach that captures inter-block nested structure in HTML documents. Section 3 introduces the approach that captures intra-block nested structure. Section 4 discusses related work. Finally we conclude in Section 5.

## 2 Capturing Inter-block Nested Structure

### 2.1 Exploiting Structural Information

We use the content of the title as the root element, and convert the content of the *meta* elements in the *head* element as the attributes of the root element. The following rule shows how to convert *head* element.

**Rule 1** Let  $D$  be an HTML document as follows:

$D = \langle \text{html} \rangle \langle \text{head} \rangle \langle \text{title} \rangle T \langle / \text{title} \rangle \langle \text{meta name} = "n_1" \text{ content} = "s_1" \dots \text{name} = "n_k" \text{ content} = "s_k" \rangle \langle / \text{head} \rangle \langle \text{body} \rangle B \langle / \text{body} \rangle \langle / \text{html} \rangle$ .

Then  $\psi(D) = \langle T \ n_1 = "s_1" \dots n_k = "s_k" \rangle \psi(B) \langle / T \rangle$ , where  $\psi$  is the converting function which converts an HTML document into XML one.

Most HTML documents use *heading* blocks to express its nested structure. We can use it to capture the inter-block nested structure as follows.

**Rule 2** Let  $D$  be an HTML section:  $D = S_0 \langle hn \rangle T_1 \langle / hn \rangle S_1 \dots \langle hn \rangle T_m \langle / hn \rangle S_m$ , where  $hn$  is the most important heading in  $D$ , and each of  $S_0, \dots, S_m$  is a sequence of HTML *blocks*. Then  $\psi(D) = \psi(S_0) \oplus \langle T'_1 \rangle \psi(S_1) \langle / T'_1 \rangle \oplus \dots \oplus \langle T'_m \rangle \psi(S_m) \langle / T'_m \rangle$ , where  $T'_i = T_i$  if  $T_i$  is a character string, and  $\langle T'_i \rangle = \langle N_i \ \text{url} = "U" \rangle$  if  $T_i = \langle a \ \text{url} = "U" \rangle N_i \langle / a \rangle$ ,  $i = 1, 2, \dots, m$ .

### 2.2 Exploiting Formatting Information

For the HTML documents in which the authors use formatting information rather than *heading* to show the nested structure, the key is to find out all the text strings that have the most important formatting information, whose function is just as the most important *heading*. Then the conversion is just the same as the conversion of the *heading* blocks. Rule 3 shows this process. The converting functions in Rule 2 and Rule 3 are recursive, we can recursively apply them to  $S_0, \dots, S_m$  till the whole nested structure in the HTML document is captured.



**Rule 3** Let  $D$  be an HTML section and  $T_1, \dots, T_m$  the character strings that have the biggest font, or in bold tags, or in strong tags. Let  $D = S_0 T_1 S_1 \dots T_m S_m$ , where  $m \geq 1$ , and each of  $S_0, \dots, S_m$  is a sequence of HTML *blocks*. Then  $\psi(D) = \psi(S_0) \oplus \langle T'_1 \rangle \psi(S_1) \langle /T'_1 \rangle \oplus \dots \oplus \langle T'_m \rangle \psi(S_m) \langle /T'_m \rangle$ , where the meaning of  $T'_i$ ,  $i = 1, 2, \dots, m$ , is the same as the one stated in Rule 2.

### 3 Capturing Intra-block Nested Structure

#### 3.1 Converting Text-Level Elements and Paragraph Elements

After capturing the inter-block nested structure, the next step is capturing the intra-block nested Structure. If a character string does not contain formatting information stated in Rule 5, then we call it a simple character string. We convert character strings and *a* (anchor) elements as follows.

**Rule 4** Let  $D = S$ . If  $S$  is a simple character string, then  $\psi(D) = S$ . If  $S$  is an *a* element:  $S = \langle a \text{ url} = "U" \rangle N \langle a \rangle$ , then if it is not the content of a heading block (i.e. h1 to h6 element) which we address in Rule 2, we convert it as an XML empty element:  $\psi(D) = \langle N \text{ url} = "U" \rangle$ .

HTML documents usually contain nested structure via formatting information including bold, italic, or colon ':' etc. We can capture the nested structure by these emphasized formatting information as follows.

**Rule 5** Let  $D = \langle t \rangle T_1 \langle t \rangle S_1 \dots \langle t \rangle T_n \langle t \rangle S_n$ , or  $D = T_1 : S_1 \dots T_n : S_n$ , where  $t$  is either **b**, **i**, **em**, or **strong**. Then  $\psi(D) = \langle T'_1 \rangle \psi(S_1) \langle /T'_1 \rangle \dots \langle T'_n \rangle \psi(S_n) \langle /T'_n \rangle$ , where the meaning of  $T'_i$ ,  $i = 1, 2, \dots, m$ , is the same as the one stated in Rule 2.

If the content of a *p* element is a simple character string, we introduce an XML element *describe*. Otherwise we just convert its content.

**Rule 6** Let  $D = \langle p \rangle S \langle /p \rangle$  or  $D = \langle p \rangle S$  (as  $\langle /p \rangle$  can be omitted in HTML), where  $S$  is a text string. If  $S$  is a simple character string, then  $\psi(D) = \langle describe \rangle S \langle /describe \rangle$ . Otherwise,  $\psi(D) = \psi(S)$

#### 3.2 Converting Lists

HTML supports unordered lists *ul*, ordered lists *ol*, and definition lists *dl*. Both unordered and ordered lists consist of *li* element, which are their items. If a *li* element is a simple character string, we introduce an XML element named *item* to mark it. Definition lists consist of two parts: a *term* given by the *dt* element, and a *description* given by the *dd* element. We summarize the process for unordered lists and ordered lists as Rule 7 and for definition lists as Rule 8.

**Rule 7** Let  $L = \langle ul \rangle L_1 \dots L_n \langle /ul \rangle$ , or  $L = \langle ol \rangle L_1 \dots L_n \langle /ol \rangle$ , where  $L_i$  is a *li* element and  $i = 1, \dots, n$ . Then  $\psi(L) = \psi(L_1) \oplus \dots \oplus \psi(L_n)$ . If the content of a *li* element is a simple character string  $S$ , then  $\psi(L_i) = \langle item \rangle S \langle /item \rangle$ .

**Rule 8** Let  $L = \langle dl \rangle R_1 \dots R_m \langle dl \rangle$ , where  $R_i = \langle dt \rangle T_i \langle /dt \rangle \langle dd \rangle D_{i1} \langle /dd \rangle \dots \langle dd \rangle D_{in_i} \langle /dd \rangle$ , and  $i = 1, 2, \dots, m$ ;  $n_i \geq 0$ . Then  $\psi(L) = \psi(R_1) \oplus \dots \oplus \psi(R_m)$ . If  $T_i$  is a simple character string, then  $\psi(R_i) = \langle T_i \rangle \psi(D_{i1}) \oplus \dots \oplus \psi(D_{in_i}) \langle /T_i \rangle$ . Otherwise  $\psi(R_i) = \psi(T_i) \oplus \psi(D_{i1}) \oplus \dots \oplus \psi(D_{in_i})$ .

### 3.3 Capturing the Attribute-Value Pairs of HTML Tables

HTML Table cells generally contain heading information via the *th* element and data via the *td* element, and may span multiple rows and columns. If a *th* or *td* element contains *colspan* =  $n$ , it means that the particular cell of the *th* or *td* is to be expanded to  $n - 1$  more columns, starting from the current cell in the current row. If a *th* or *td* element contains *rowspan* =  $n$ , it means the particular cell of the *th* or *td* element is to be expanded to the next  $n - 1$  rows in the current column. By inserting redundant cells according to the attributes *colspan* and *rowspan*, we can normalize an HTML table in the form of Table 1.

**Table 1.** The normalized HTML table

			$h_{1,q+1}$	$\dots$	$h_{1,n}$
			$\vdots$		$\vdots$
			$h_{p,q+1}$	$\dots$	$h_{p,n}$
$v_{p+1,1}$	$\dots$	$v_{p+1,q}$	$d_{p+1,q+1}$	$\dots$	$d_{p+1,n}$
$\vdots$		$\vdots$	$\vdots$		$\vdots$
$v_{m,1}$	$\dots$	$v_{m,q}$	$d_{m,q+1}$	$\dots$	$d_{m,n}$

Based on the normalized table Table 1, we introduce a mapping rule to map the attribute-value pairs of HTML tables to the corresponding XML documents, and a merging rule to merge the content of the same XML element as Rule 9, which covers not only two dimensional tables but also one dimensional tables.

**Rule 9** Let the normalized table of an HTML table  $T$  be Table 1, the content of its *caption* be  $c$ , and the rows that contain data be  $r_{p+1}, r_{p+2}, \dots, r_m$ , where  $0 \leq p \leq m$ . Then  $\psi(T) = \langle c \rangle \psi(r_{p+1}) \oplus \psi(r_{p+2}) \oplus \dots \oplus \psi(r_m) \langle /c \rangle$ , where

$$\begin{aligned}
 \psi(r_i) = & \langle v_{i,1} \rangle \dots \langle v_{i,q} \rangle \\
 & \langle h_{1,q+1} \rangle \dots \langle h_{p,q+1} \rangle \psi(d_{i,q+1}) \langle /h_{p,q+1} \rangle \dots \langle /h_{1,q+1} \rangle \\
 & \vdots \\
 & \langle h_{1,n} \rangle \dots \langle h_{p,n} \rangle \psi(d_{i,n}) \langle /h_{p,n} \rangle \dots \langle /h_{1,n} \rangle \\
 & \langle /v_{i,q} \rangle \dots \langle /v_{i,1} \rangle, \\
 & \text{where } i = p + 1, \dots, m; \ 0 \leq p \leq m; \ 0 \leq q \leq n.
 \end{aligned}$$

Merging rule:  $\langle t1 \rangle \langle t2 \rangle s2 \langle /t2 \rangle \langle /t1 \rangle$   
 $\quad \langle t1 \rangle \langle t3 \rangle s3 \langle /t3 \rangle \langle /t1 \rangle$   
 $= \langle t1 \rangle \langle t2 \rangle s2 \langle /t2 \rangle \langle t3 \rangle s3 \langle /t3 \rangle \langle /t1 \rangle$

### 3.4 Recognizing Headings of HTML Tables

In HTML documents, there are many HTML tables without marked headings via *th* elements. To convert them by using Rule 9, we need to recognize their headings. Generally, authors use different formatting information, which are mainly font (size, bold), to mark headings for visual easy recognition. So we introduce a notion *eigenvalue* to measure the difference of formatting information.

**Definition 1** In an HTML table, if the cell's font size is  $n$ , then its *eigenvalue*  $\lambda = n$ ; moreover, if the cell has bold font, then  $\lambda = \lambda + 1$ . The *eigenvalue* of a row or a column is the *average eigenvalue* of all the cells in the row or column.

Based on the fact that the headings generally have bigger *eigenvalue* than the data part. We can recognize headings of HTML tables as follows.

**Rule 10** Let an HTML table have  $m$  rows, and the eigenvalues of  $m$  rows be  $\lambda_1, \dots, \lambda_m$ , respectively. Let the difference of two adjacent rows be  $\lambda_{12}, \lambda_{23}, \dots, \lambda_{(m-1)m}$ , where  $\lambda_{i(i+1)} = \lambda_i - \lambda_{i+1}$ ,  $i = 1, \dots, m-1$ . Let  $k$  be the minimum that  $\lambda_{k(k+1)} = \max(\lambda_{12}, \lambda_{23}, \dots, \lambda_{(m-1)m})$  and  $\lambda_{k+1} < \lambda_k$  hold. Then the first  $k$  rows are the column headings. The recognizing of row headings is just the same as of column headings.

## 4 Related Work

To capture the nested structure in HTML documents, the manual approaches [3, 6] often involve the writing of complicated code. Lixto [10] manage to avoid the writing of code by providing a fully visual and interactive user interface. The induction systems [1, 2] need to be trained with some examples, under human supervision. In [5], an ontology-based approach are proposed, which also involves the writing of complicated code, such as RDF. [7] present a case-based, but it only cover a portion of HTML pages. In [2], a semi-automatic wrapper generation approach is presented that only exploits formatting information to hypothesize the underlying structure of a page, so that it cannot correctly convert complex *table* and *list*. In [4], a heuristic approach for automatic conversion is introduced. However, it only uses the structural information, neglecting the formatting information, so that it cannot convert the HTML documents that use formatting information to mark its nested structure. Since HTML tables are used frequently in HTML documents and are information rich, they gain a lot of attentions recently [8, 9]. In [8], an automatic approach that extracts attribute-value pairs is presented. But it gives no formal approach to capture the attribute-value pairs of HTML tables with nested headings, and cannot capture the nested structure in cells of HTML tables.

In this paper, we introduce a new approach to automatically convert HTML documents into XML documents. A system based on the approach presented in this paper has been implemented. We must stress the fact that some of the presented rules are based on heuristics and might fail. Compared with the other

related approaches, as our approach exploits both formatting and structural information implied by HTML tags to automatically capture the nested structure in HTML documents, and introduces a mapping rule to automatically map the attribute-value pairs in HTML tables to XML documents and a heuristic rule to recognize headings of HTML tables by formatting information, it improves the expressive capability of the existing approaches.

## 5 Conclusion

The main contribution of this paper is that we present a new approach that automatically captures the nested structure in HTML documents and converts it into XML ones. Although capturing the nested structure in HTML documents is difficult, as our approach exploits both the formatting information and the structural information implied by HTML tags, it improves the expressive capability of the existing approaches. It can be used as an effective auxiliary tool in recycling HTML documents as XML documents.

## Acknowledgements

This research is supported by NSFC (National Natural Science Foundation of China) (60173045).

## References

1. Nicholas Kushmerick, D. Weld, and R. Doorenbos. Wrapper Induction for Information Extraction. In *IJCAI* (1997), 729–737
2. Naveen Ashish and Craig A. Knoblock. Semi-Automatic Wrapper Generation for Internet Information Sources. In *CoopIS* (1997), 160–169
3. Joachim Hammer, Hector Garcia-Molina, Svetlozar Nestorov, Ramana Yerneni, Markus M. Breunig, and Vasilis Vassalos. Template-Based Wrappers in the TSIM-MIS System. In *SIGMOD Conference* (1997), 532–535.
4. Seung Jin Lim and Yiu-Kai Ng. A Heuristic Approach for Coverting HTML Documents to XML Documents. In J.Loyd et al., editor, *CL* (2000), 1182–1196
5. Thomas E. Potok, Mark T. Elmore, Joel W. Reed, and Nagiza F. Samatova. An Ontology- Based HTML to XML Conversion Using Intelligent Agents. In J.Loyd et al., editor, *HICSS* (2002), 120–129.
6. Arnaud Sahuguet and Fabien Azavant. Building Intelligent Web Applications Using Lightweight Wrappers. *Data and Knowledge Engineering* (2001), 36(3): 283–316
7. Masayuki Umehara, Koji Iwanuma, and Hidetomo Nabeshima. A Case-Based Recognition of Semantic Structures in HTML Documents. In *IDEAL* (2002), 141–147
8. Yingchen Yang and Wo-Shun Luk. A Framework for Web Table Mining. In *WIDM'02* (2002), 36–42
9. Shijun Li, Mengchi Liu, Guoren Wang, and Zhiyong Peng. Capturing Semantic hierarchies to Perform Meaningful Integration in HTML Tables. In *APWEB* (2004), 899–902
10. R. Baumgartner, S. Flesca and G. Gottlob. Visual Web Information Extraction with Lixto. In *VLDB* (2001), 119–128

# Quality Improvement Modeling and Practice in Baosteel Based on KIV-KOV Analysis

Xinyu Liu<sup>1,2</sup>, Haidong Tang<sup>3</sup>, Zhiping Fan<sup>1</sup>, and Bing Deng<sup>2</sup>

<sup>1</sup> School of Business & Administration, Northeastern University, Shenyang 110004, P.R.C

<sup>2</sup> Baoshan Iron & Steel Co., Ltd. Shanghai 201900, P.R.China

<sup>3</sup> Data Strategies Dept., Shanghai Baosight Software Co., Ltd. Shanghai 201900, P.R.China  
liuxy@baosteel.com

**Abstract.** The conventional manufacture process improvement is a long working process which was based on personal experience and remains a questionable cost effective issue. It really demands a fast and low cost manufacture process design and improvement model. The article indicates that the core process of quality improvement is to define the relationship of KIV-KOV and to optimize. With the data warehouse and data mining techniques, this article offers the KIV-KOV based new quality improvement model, functional structure and process as well as the practical experience.

## 1 Introduction

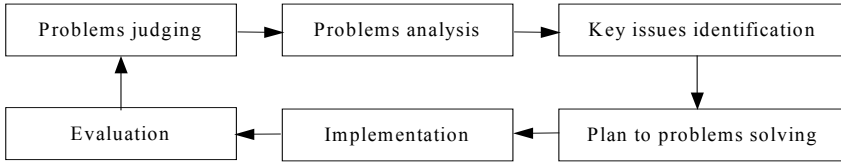
Along with maturity of manufacture control system, a great number of online manufacture data can be collected, analyzed and mined to discover the internal relationship between technique parameter and product performance. Hence it can upgrade the manufacture process to improve product quality and meet the customer absolute requirement on quality improvement.

## 2 Quality Improvement Model

### 2.1 Quality Improvement Model Based on KIV-KOV Analysis

The core process of control model is to identify key reasons of fault quality by way of KIV-KOV analysis (KIV: Key Input Variable; KOV: Key Output variable). Normally engineers only conduct quite few KIV-KOV analyses based on their personal experience rather than vast analysis due to the complexity of calculation. The more advanced quality control model based on data warehouse and data mining techniques can realize a great number of KIV-KOV analyses by implementing computerized statistic analysis tools. It can conduct analysis in either overall process or up-down process depends on the complexity of problem.

Process of manufacture technique design and upgrade based on PDCA cycle is shown in Fig.1.



**Fig. 1.** Process of manufacture technique design and upgrade based on PDCA cycle

The actual process has been shown in the followings,

- Step1: Lodge problem: define the target variable KOV according to the actual quality problem;
- Step2: Analyze problem: identify the relevant quality factors IV according to target;
- Step3: Identify key issue: identify the key factors e.g. KIV by using data mining technique to analyze relevant data;
- Step4: Problem solving planning: optimize manufacture techniques control parameter by supporting of data mining technology;
- Step5: Implementation: implement the optimized process into practice;
- Step6: Evaluation: evaluate the correctness and reasonableness otherwise back to second step.

## 2.2 Key Process of Product Quality Improvement

The core process of quality improvement is to find out the relationship between KIV-KOV. Data mining is the key business intelligence technology based on vast amount of data. Generally, the most time consumed thing in data mining is data collection and these data need to be presorted based on desired information objectives and even be sampled if it is a large scale of data. Data has to be properly organized for effective information processing. It is easier to sort data because data has been tested its consistency before entering data warehouse.

When conducting multiprocessor computing, it is more difficult to process the data because of database format, data format and other mistakes. It can be further investigated to better understand data by implementing proper presort such as key variable select, rank of importance, diagnosis of abnormal node and conflicting data.

Cluster control and modeling are hardcore in data mining process in Baosteel. Ward method and K means are main methods of cluster control. Neural networks, confusion matrix and liner regression are all core modeling techniques.

Generally, the goal of data mining in Baosteel is to optimize one specific target function, e.g. tensile strength, which can be better from the optimization modeling.

It can be summarized that the basic processes of product quality improvement as following 4 sequential steps: data collection from data warehouse, data pre-sorting, modeling, and optimization.

3 KIV-KOV Analysis Technologies in Quality Improvement System

Quality improvement system can be built up based on data warehouse and data mining technologies by integration of ten modules as following.

Data collection module consists of data format and data mining.

Data sampling module creates a subset of data from the vast amount whole by choosing the sample through preset criteria.

Data presort module is very important to presort data for better understanding of data. Data presort system includes standardized management, variable screen, variable transforming, diagnosis of abnormal node and conflicting data.

Visualized exploration module can provide following functions, including distribution diagram, scatter diagram and correlation analysis.

Cluster control module can provide multi-cluster methods for users to cluster observed data and understand the distribution of “up grade” or “low grade”.

Modeling module can provide three modeling techniques. Neural network provides MLP and RBF modeling further to optimize the neural network structure through advanced computation. Confusion matrix provides default and advanced computation based confusion matrix. Linear regression modeling provides 4 modeling techniques including all variable regression, gradual regression, front regression and back regression.

Data prediction module can predict KIV observed data.

Optimization module provides users methods of quality improvement

Trends control module can show one variable trends of change in graphic view with a defined time duration change. Users can draw scatter diagram or trend diagram by using day, month or quarter as a unit.

Standardized management module provides users to set changeable range by defining variable.

Process procedure and internal control are presented in Table 1.

Table 1. System implementation steps practice in KIV-KOV analysis

Setp1	Setp2	Setp3	Setp4	Setp5	Setp6	Setp7
Raw data	Sampling	Observation	Modulating	Modeling	Evaluation	Optimization
DBF	Data selection	Windows	Windows	Neural networks	Deviation control	Optimize screen
ORACLE	Data transforming	Visualization	Control of abnormal node	Confusion matrix	Reporting	GA method
DB2	Sampling methods		Data screen	Regression control		Optimize structure
SAS			Variable screen			Visualization
Text			Rank of importance			Individual factor control

4 KIV-KOV Analysis Technologies Application in Quality Improvement for Baosteel

In the long period, Baosteel adopt the chemistry design of ‘high manganese and low carbon’ to produce the hot rolling plate used by welding pipe which steel grade is SP. In recent years as we communicate with other corporations, we found that the content of Manganese of the same steel grade of other corporations is lower than ours, but its mechanical property is better than ours. The discovery attracts our attention. So we start to study the chemistry design and process of SP deeply.

4.1 Investigation

Elongation and tensile strength are important index of hot rolling plate. They will directly influence the customer’s use. We investigate the elongation and tensile strength of SP produced in recent years (See Fig.2).

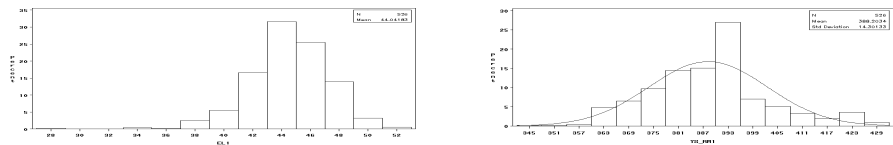


Fig. 2. Histogram of elongation and tensile strength

The figure shows that the average value of elongation is 44 and the average value of tensile strength is 388 Mega-pa. Although these values can satisfy the request of enterprise standard, they are lower than other corporation’s product.

According to the condition we have mastered, we investigate the factors which would influence the mechanical property. The factors include the content of carbon, manganese, phosphor, silicon, sulfur, post-rolling temperature, coil temperature(See Table 2). We want to adjust the factor and improvement the quality of steel plate.

Table 2. Factors

Variable	Label	N	Mean	Std Dev	Minimum	Maximum
C_EL	C_EL	510	84.2294118	14.7405965	0	163.0000000
MN_EL	MN_EL	510	40.6784314	4.9100480	0	48.0000000
SI_EL	SI_EL	510	14.9372549	4.1843548	0	53.0000000
P_EL	P_EL	510	15.4156863	4.2853778	0	29.0000000
S_EL	S_EL	510	71.8882353	26.0597255	0	150.0000000
FT_AVER	FT_AVER	526	856.3517110	6.3530920	842.0000000	895.0000000
CT_AVER	CT_AVER	526	622.4904943	61.7079674	0	764.0000000

4.2 KIV-KOV Analysis

The KIV-KOV analysis step is presented as below.

Step1: Data preparing: We select the data of the common carbon steel produced by 1580 hot rolling plants from quality data warehouse. At this time we pay attention that the data should include several steel grade and the contents of carbon and manganese should vary at large scale.



- Step2: Designating the target variable: We designate elongation and tensile strength as target variable. Meanwhile, we get rid of the irrelative variable.
- Step3: Partition: We split the data into three parts. 70% for training, 20% for validation and the rest for model testing.
- Step4: Building, training and validation of regression & neural network model.
- Step5: Assessment and conclusion comparison: We use 545 records to testing the regression model and the neural network model. Both of their accuracy is above 80% and the network model is better than the regression model.
- Step6: Selecting model: neural network model as final model to analysis the data.

### 4.3 Analysis and Results

According to the result of the neural network model, we emphasize the study of the influence between the contents of carbon, manganese, coiling temperature and mechanical property. We discover that

- 1) If the content of manganese increase, the value of elongation decrease and tensile strength increase(See Fig.3).

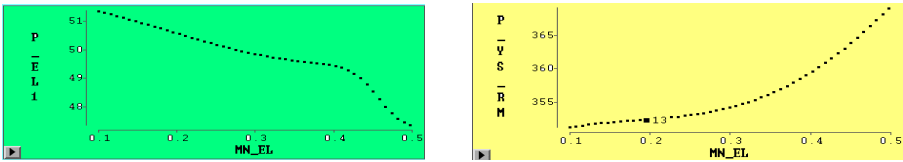


Fig. 3. The relation between the content of manganese and mechanical property

- 2) If the content of carbon increase, the value of elongation decrease and tensile strength increase(See Fig.4).

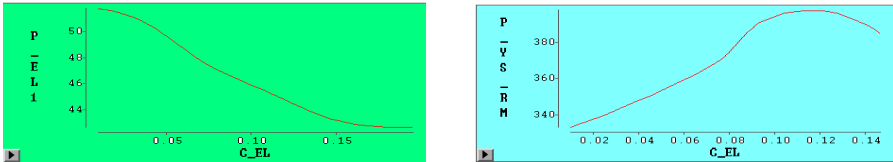


Fig. 4. The relation between the content of carbon and mechanical property

- 3) If the coiling temperature increases, the value of elongation increase and tensile strength decrease(See Fig.5).

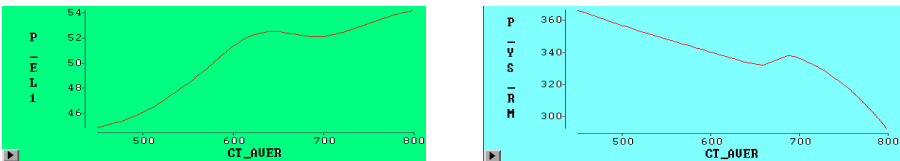


Fig. 5. The relation between the coiling temperature and mechanical property

4.4 Measures and Effects

We discover that the content of manganese and carbon has the same influence upon the mechanical property, but the price of carbon is cheaper than manganese. So we decided to decrease the content of manganese and increase the content of carbon, then we adjust the coiling temperature. After a period of testing we get the production which is better than the original SP(see Table 3).

The more important thing is that we decrease the cost of product largely. The cost of per ton is less 7 RMB YUAN than before and it will save 1.6 million RMB YUAN every year for our corporation.

Table 3. The mechanical property compared new process and old process

Mechanical property	New process	Old process	Standard
Elongation	47	44	≥35
tensile strength	385	390	≥440

5 Conclusions

As the coming of information times we can get millions of data. How to use data is an important problem that everyone has to face. In normal case, engineers only conduct quite few KIV-KOV analyses based on their personal experience rather than vast analysis due to the complexity of calculation. This system, which provide the more advanced quality control model based on data warehouse and data mining techniques, can realize a great number of KIV-KOV analysis by implementing computerized statistic analysis tools. It can conduct analysis in either overall process or up-down process depends on the complexity of problem.

References

1. Jiawei Han, Micheline Kamber: Data Mining: Concepts and Techniques [M], High Education Publishing House 2001
2. Ge Yu, Yubin Bao, Xiao Ji: Data Warehousing Methodology [M], Northeastern University Publishing House. 2003
3. Deng Bing, Development of the Baosteel Practical Data Mining System and Study on Clustering Algorithms for Product Quality Control [D]. Northeastern University, 2002.

# A Frequent Pattern Discovery Method for Outlier Detection\*

Zengyou He<sup>1</sup>, Xiaofei Xu<sup>1</sup>, Joshua Zhexue Huang<sup>2</sup>, and Shengchun Deng<sup>1</sup>

<sup>1</sup> Department of Computer Science and Engineering, Harbin Institute of Technology, China  
zengyouhe@yahoo.com, {xiaofei,dsc}@hit.edu.cn

<sup>2</sup> E-Business Technology Institute, The University of Hong Kong, Pokfulam, Hong Kong  
jhuang@eti.hku.hk

**Abstract.** An outlier in a dataset is an observation or a point that is considerably dissimilar to or inconsistent with the remainder of the data. Detection of outliers is important for many applications and has recently attracted much attention in the data mining research community. In this paper, we present a new method to detect outliers by discovering frequent patterns (or frequent itemsets) from the data set. The outliers are defined as the data transactions which contain less frequent patterns in their itemsets. We define a measure called *FPOF* (*Frequent Pattern Outlier Factor*) to detect the outlier transactions and propose the *FindFPOF* algorithm to discover outliers. The experimental results show that our approach outperformed the existing methods on identifying interesting outliers.

## 1 Introduction

An outlier is an observation that deviates so much from other observations as to arouse suspicion that it was generated by a different mechanism [12]. Mining outliers from data is an important data mining task and required in many real applications, such as credit card fraud detection, discovery of criminal activities in electronic commerce, weather prediction, marketing and customer segmentation.

Recently, methods for finding such outliers in large datasets have drawn increasing attention [e.g., 1-7]. In this paper, we present a new method for detecting outliers by discovering the frequent itemsets. The basic idea is very simple. Since the frequent itemsets (we call them frequent patterns in this paper) discovered by the association rule algorithm [8] reflect the “*common patterns*” in the dataset, it is reasonable and intuitive to regard those data points which contain less frequent patterns as outliers. In other words, if a data object contains more frequent patterns, it means that this data object is unlikely to be an outlier because it possesses the “*common features*” of the dataset. Those frequent patterns that are not contained in the data objects can be used

---

\* The High Technology Research and Development Program of China (No. 2002AA413310, No. 2003AA4Z2170, No. 2003AA413021) and the IBM SUR Research Fund supported this research.

as *descriptions* of outliers in data. We define a measure called *FPOF* (*Frequent Pattern Outlier Factor*) to detect the outlying objects and propose the *FindFPOF* algorithm to discover them from the data.

## 2 Proposed Method

Agrawal formulated the problem of discovering frequent itemsets in market basket databases as follows [8]:

Let  $I = \{i_1, i_2, \dots, i_m\}$  be a set of  $m$  literals called *items* and the database  $D = \{t_1, t_2, \dots, t_n\}$  a set of  $n$  transactions, each consisting of a set of items from  $I$ . An itemset  $X$  is a non-empty subset of  $I$ . The length of itemset  $X$  is the number of items in  $X$ . An itemset of length  $k$  is called a  $k$ -*itemset*. A transaction  $t \in D$  is said to contain itemset  $X$  if  $X \subseteq t$ . The *support* of itemset  $X$  is defined as  $\text{support}(X) = \|\{t \in D \mid X \subseteq t\}\| / \|\{t \in D\}\|$ .

The problem of finding all *frequent itemsets* in  $D$  is defined as follows. Given a user defined threshold *minisupport*, find all itemsets with supports greater or equal to *minisupport*. Frequent itemsets are also called *frequent patterns*.

From the viewpoint of knowledge discovery, frequent patterns reflect the “*common patterns*” that apply to many objects, or to large percentage of objects in the dataset. In contrast, outlier detection focuses on a very small percentage of data objects. Hence, the idea of making use of frequent patterns for outlier detection is very intuitive.

**Definition 1.** (*FPOF-Frequent Pattern Outlier Factor*) Let  $D = \{t_1, t_2, \dots, t_n\}$  be a database containing a set of  $n$  transactions with items  $I$ . Given a threshold *minisupport*, the set of all frequent patterns is denoted as:  $FPS(D, \text{minisupport})$ . For each transaction  $t$ , the *Frequent Pattern Outlier Factor* of  $t$  is defined as:

$$FPOF(t) = \frac{\sum_X \text{support}(X)}{\|FPS(D, \text{minisupport})\|}, \text{ where } X \subseteq t \text{ and } X \in FPS(D, \text{minisupport}) \quad (1)$$

The interpretation of formula (1) is as follows. If a transaction  $t$  contains more frequent patterns, its *FPOF* value will be big, which indicates that it is unlikely to be an outlier. In contrast, transactions with small *FPOF* values are likely to be outliers. The *FPOF* value is between 0 and 1.

**Definition 2.** For each transaction  $t$ , an itemset  $X$  is said to be *contradictive* to  $t$  if  $X \not\subseteq t$ . The *contradict-ness* of  $X$  to  $t$  is defined as:

$$\text{Contradict-ness}(X, t) = (\|X\| - \|t \cap X\|) * \text{support}(X) \quad (2)$$

In our approach, the *frequent pattern outlier factor* given in Definition 1 is used as the basic measure for *identifying* outliers. To *describe* the reasons why identified outliers are abnormal, the itemsets not contained in the transaction (it is said that the itemset is *contradictive* to the transaction) are good candidates for describing the reasons.

The consideration behind formula (2) is as follows. Firstly, the greater the support of the itemset  $X$ , the greater the value of *contradict-ness* of  $X$  to  $t$ , since a large support value of  $X$  suggests a big deviation. Secondly, longer itemsets give better description than that of short ones.

With definition 2, it is possible to identify the contribution of each itemset to the outlying-ness of the specified transaction. However, since it is not feasible to list all the *contradict itemsets*, it is preferable to present only the *top k contradict frequent patterns* to the end user, as given in Definition 3 below.

**Definition 3.** (*TKCFP-Top K Contradict Frequent Pattern*) Given  $D$ ,  $I$ , *minisupport* and  $FPS(D, \text{minisupport})$  as defined in Definition 1. For each transaction  $t$ , the itemset  $X$  is said to be a *top k contradict frequent pattern* if there exist no more than  $(k-1)$  itemsets whose *contradict-ness* is higher than that of  $X$ , where  $X \in FPS(D, \text{minisupport})$ .

Our task is to mine *top-n* outliers with regard to the value of *frequent pattern outlier factor*. For each identified outlier, its *top k contradict frequent patterns* will also be discovered for the purpose of description.

**Algorithm FindFPOF**

```

Input:    $D$            // the transaction database
           $\text{minisupport}$  // user defined threshold for the permissible minimal support
           $\text{top-n}$         // user defined threshold value for top n fp-outliers
           $\text{top-k}$         // user defined threshold value for top k contradict frequent patterns

Output: The values of FPOF for all transactions // indicates the degree of deviation
          The top-n FP-outliers with their corresponding TKCFPs

01 begin
02     Mining the set of frequent patterns on database  $D$  using  $\text{minisupport}$ 
03     /* the set of all frequent patterns is donated as:  $FPS(D, \text{minisupport})$  */
04     foreach transaction  $t$  in  $D$  do begin
05         foreach frequent pattern  $X$  in  $FPS(D, \text{minisupport})$  do begin
06             if  $t$  contains  $X$  then
07                  $FPOF(t) = FPOF(t) + \text{support}(X)$ 
08             return  $FPOF(t)$ 
09     Output the transactions in the ascending order of their FPOF values. Stop when it outputs top-n
    transactions
10     foreach transaction  $t$  in top-n outliers do begin
11         Finds its top-k contradict frequent patterns and outputs them

16 end

```

**Fig. 1.** The FindFPOF Algorithm

### 3 Algorithm for Detecting FP-Outliers

Using the outlier factor *FPOF*, we can determine the degree of a record's deviation. In this section, we present our algorithm for detecting outliers.

The algorithm *FindFPOF* for detecting outliers is listed in Fig. 1. The algorithm first gets the frequent patterns from the database using an existing association rule

mining algorithm with a given *minisupport* (Step 2-3). Then, for every transaction in the database, the value of *FPOF* is computed according to Definition 1 (Step 4-9). Finally, the *top-n* FP-outliers are output with their corresponding *top-k* contradict frequent patterns (Step 10-11).

The *FindFPOF* algorithm has three parts: 1) mining the frequent patterns from the database; 2) computing the value of *FPOF* for each transaction; and 3) finding the *top-n* FP-outliers with their *TKCFP* descriptions. The computational cost of the frequent-pattern mining algorithm is donated as  $O(FP)$ . We remark that many fast frequent-pattern mining algorithms are available [e.g., 8-9] and so the computation complexity of Part 1 will be acceptable. As to Part 2, two “for loops” are required. Therefore, the computational cost of this part is  $O(N*S)$ , where  $N$  is number of the transactions in the database and  $S$  is the size of the set of frequent patterns. Part 3 has the computational cost of  $O(N*\log N + S*(top-n)*(top-k)*\log(top-k))$ , because finding the *top-n* outliers and the *top-k* contradict frequent patterns for each identified outlier needs a *sort* operation.

The overall computational complexity of the *FindFPOF* algorithm is:

$$O(FP + N*S + N*\log N + S*(top-n)*(top-k)*\log(top-k)) \quad (3)$$

## 4 Experimental Results

### 4.1 Experiment Design and Evaluation Method

We used two real datasets from UCI [10] to demonstrate the effectiveness of our algorithms against the *FindCBLOF* algorithm [7] and the *KNN* algorithm [2]. For all the experiments, the two parameters needed by the *FindCBLOF* algorithm were set to 90% and 5 separately as done in [7]. For the *KNN* algorithm [2], the results were obtained using the *5-nearest-neighbours*. The results didn't change significantly when the parameter  $k$  was specified to alternative values. For our algorithm, the parameter *minisupport* for mining frequent patterns was fixed to 10%, and the maximal number of items in an itemset was set to 5.

As pointed out by Aggarwal and Yu [4], one way to test how well the outlier detection algorithm worked is to run the method on the dataset and test the percentage of points which belong to the rare classes. If outlier detection works well, it is expected that the rare classes would be over-represented in the set of points found. These kinds of classes are also interesting from a practical perspective.

Since we knew the true class of each object in the test dataset, we defined the objects in small classes as rare cases. The number of rare cases identified was used as the assessment basis for comparing our algorithm with other algorithms.

### 4.2 Lymphography Data

The first dataset used was the Lymphography data set, which had 148 instances with 18 attributes. Among these 18 attributes, there were some numeric attributes. We discretized the numeric attributes using the automatic discretization functionality

provided by the CBA [11] software. The data set contained a total of 4 classes. Classes 2 and 3 had the largest number of instances. The remained classes were regarded as rare class labels. The corresponding class distribution is illustrated in Table 1.

**Table 1.** Class Distribution of Lymphography Data Set

Case	Class codes	Percentage of instances
Commonly Occurring Classes	2,3	95.9%
Rare Classes	1,4	4.1%

Table 2 shows the results produced by the *FindFPOF* algorithm against the *FindCBLOF* and *KNN* algorithms. Here, the *top ratio* is ratio of the number of records specified as *top-k* outliers to that of the records in the dataset. The *coverage* is ratio of the number of detected rare classes to that of the rare classes in the dataset. For example, we let the *FindFPOF* algorithm find the *top 16* outliers with the top ratio of 11%. By examining these 16 points, we found that 6 of them belonged to the rare classes. In contrast, when we ran the *FindCBLOF* algorithm on this dataset, we found that only 4 of 16 top outliers belonged to rare classes.

**Table 2.** Detected Rare Classes in Lymphography Dataset

Top Ratio (Number of Records)	Number of Rare Classes Included (Coverage)		
	<i>FindFPOF</i>	<i>FindCBLOF</i>	<i>KNN</i>
5% (7)	<b>5(83%)</b>	4 (67%)	1 (17%)
10%(15)	<b>5(83%)</b>	4 (67%)	1 (17%)
11%(16)	<b>6(100%)</b>	4 (67%)	1 (17%)
15%(22)	<b>6 (100%)</b>	4 (67%)	2 (33%)
20%(30)	<b>6 (100%)</b>	<b>6 (100%)</b>	2 (33%)

Furthermore, in this experiment, the *FindFPOF* algorithm performed the best for all cases and could find all the records in rare classes when the *top ratio* reached 11%. In contrast, the *FindCBLOF* algorithm achieved this goal with the *top ratio* at 20%, which was almost the twice for that of our algorithm.

**4.3 Wisconsin Breast Cancer Data**

The second dataset used was the Wisconsin breast cancer data set, which had 699 instances with 9 attributes. In this experiment, all attributes were considered as categorical. Each record was labeled as *benign* (458 or 65.5%) or *malignant* (241 or 34.5%). We followed the experimental method of Harkins, et al. [6] by removing some of the *malignant* records to form a very unbalanced distribution. The resultant dataset had 39 (8%) *malignant* records and 444 (92%) *benign* records<sup>1</sup>. The corresponding class distribution is illustrated in Table 3.

<sup>1</sup> The resultant dataset is public available at:  
<http://research.cmis.csiro.au/rohanb/outliers/breast-cancer/>

**Table 3.** Class Distribution of Wisconsin breast cancer data set

Case	Class codes	Percentage of instances
Commonly Occurring Classes	1	92%
Rare Classes	2	8%

With this dataset, our aim was to test the performance of our algorithm against the *FindCBLOF* algorithm, the *KNN* algorithm and the *RNN* based outlier detection algorithm [6]. The results of the *RNN* based outlier detection algorithm on this dataset were reported in [6].

Table 4 shows the results produced by the 4 algorithms. One important observation from Table 4 was that, comparing with the *KNN* and *RNN* algorithms, our algorithm performed the best for all cases and never performed the worst. That is to say, the *FindFPOF* algorithm was more capable to effectively detect outliers than the other two algorithms. In comparison to the *FindCBLOF* algorithm, our algorithm achieved the same average performance with respect to the number of outliers identified.

Another important observation was that the *FindFPOF* algorithm found all the *malignant* records with the *top ratio* at 14%. In contrast, the *RNN* based outlier detection algorithm achieved this goal with the *top ratio* at 28%, which was the twice for that of *FindFPOF*, while the *FindCBLOF* algorithm achieved this goal with the *top ratio* at 16%.

**Table 4.** Detected Malignant Records in Wisconsin Breast Cancer Dataset

Top Ratio (Number of Records)	Number of Malignant Included (Coverage)			
	<i>FindFPOF</i>	<i>FindCBLOF</i>	<i>RNN</i>	<i>KNN</i>
1%(4)	3(7.69%)	4 (10.26%)	3 (7.69%)	4 (10.26%)
2%(8)	7 (17.95%)	7 (17.95%)	6 (15.38%)	4 (10.26%)
4%(16)	14 (35.90%)	14 (35.90%)	11 (28.21%)	9 (23.80%)
6%(24)	21 (53.85%)	21 (53.85%)	18 (46.15%)	15 (38.46%)
8%(32)	28(71.79%)	27 (69.23%)	25 (64.10%)	20 (51.28%)
10%(40)	31(79.49%)	32 (82.05%)	30 (76.92%)	21 (53.83%)
12%(48)	35 (89.74%)	35 (89.74%)	35 (89.74%)	26 (66.67%)
14%(56)	39(100.00%)	38 (97.44%)	36 (92.31%)	28 (71.79%)
16%(64)	39(100.00%)	39(100.00%)	36(92.31%)	28(71.79%)
18%(72)	<b>39 (100.00%)</b>	<b>39 (100.00%)</b>	38 (97.44%)	28 (71.79%)
20%(80)	<b>39 (100.00%)</b>	<b>39 (100.00%)</b>	38 (97.44%)	28 (71.79%)
25%(100)	<b>39 (100.00%)</b>	<b>39 (100.00%)</b>	38 (97.44%)	28 (71.79%)
28%(112)	<b>39 (100.00%)</b>	<b>39 (100.00%)</b>	39(100.00%)	28 (71.79%)

## 5 Conclusions

Frequent pattern mining and outlier detection are two integral parts of data mining and have attracted attentions in their own fields. Based on frequent patterns, this paper has proposed a new outlier detection method. The effectiveness of our method was verified by the experimental results.



## References

- [1] E. M. Knorr, R. T. Ng. Algorithms for Mining Distance-Based Outliers in Large Datasets. VLDB'98, 1998.
- [2] S. Ramaswamy, et al. Efficient Algorithms for Mining Outliers from Large Data Sets. SIGMOD'00, 2000.
- [3] M. M. Breunig, et al. LOF: Identifying Density-Based Local Outliers. SIGMOD'00, 2000.
- [4] C. Aggarwal, P. Yu. Outlier Detection for High Dimensional Data. In: SIGMOD'01, 2001.
- [5] L. Wei, W. Qian, A. Zhou, W. Jin, J. X. Yu. HOT: Hypergraph-Based Outlier Test for Categorical Data. PAKDD'03, 2003
- [6] S. Harkins, H. He, G. J. Williams, R. A. Baster. Outlier Detection Using Replicator Neural Networks. DaWaK'02, 2002.
- [7] Z. He, X. Xu, S. Deng. Discovering Cluster Based Local Outliers. Pattern Recognition Letters, 2003.
- [8] R. Agrawal, R. Srikant. Fast Algorithms for Mining Association Rules. VLDB'94, 1994.
- [9] J. Han, J. Pei, J. Yin. Mining Frequent Patterns without Candidate Generation. SIGMOD'00, 2000.
- [10] G. Merz, P. Murphy. Uci repository of machine learning databases.: <http://www.ics.uci.edu/mllearn/MLRepository.html>, 1996.
- [11] B. Liu, W. Hsu, and Y. Ma. Integrating classification and association rule mining. KDD'98, 1998.
- [12] D. Hawkins. Identification of outliers. Chapman and Hall, Reading, London, 1980.

# Adaptive Load Balancing and Fault Tolerance in Push-Based Information Delivery Middleware Service

Zhaofeng Ma and Boqin Feng

Department of Computer Science and Technology,  
Xi'an Jiaotong University, Xi'an, 710049, China  
supermzf@mail.china.com

**Abstract.** Push-based personalized information delivery is a special kind of event-driven application, which must monitor or react to changes of information sources or users interests to provide user-centered real-time services. In this paper, an enhanced event-driven middleware framework of Quality of Service (QoS) is proposed to support high reliable and available timely information push-delivery, in which adaptive loading balance strategy that combines static and dynamic load distribution algorithms together to ensure the system works in a reasonable workload level. As for the fault tolerance, a super-demon with adaptable time-piece is adopted for failure detection, node replication and recovery. Large amount simulations showed the adaptable QoS strategies and algorithms that we proposed are effective, efficient for event-driven distributed systems.

## 1 Introduction

With the explosive increase of Internet applications, it is tedious work for users to seek for kinds of information sources, push-based information delivery become a promising way to serve users in an active way and provide user-centered information. Unlike pull-based information, the push-based information service works in a server-initiated way, for large amount of users, the most important issued should be taken into account is QoS. Load balancing and fault tolerance are two most important topics of QoS in distributed system, which are associated closely. Static and dynamic load balancing algorithms are two major ones for load balancing, where dynamic load balancing algorithms can be classified into sender-initiated, receiver-initiated and symmetrically-initiated algorithms [1,5,7], Thus, most load balancing algorithms can be structured as composing of a sender-initiated component and a receiver-initiated component, additional with other standard components. While failure can be classified according to their severity as: fail-stop, omission to their send/receive message, general omission of messages, and Byzantine failure [1,5,7,8]. Most practical fault tolerance algorithms are based on a timeout mechanism to detect, recovery, and replicate the failed node. To ensure that the system can resume to a correct state after a failure, or the replacement of a fail-stop processor, recovery algorithms are implemented often based on various types of checkpointing algorithms [3,5,8].

Upon the Push-based personalized information delivery system, CORBA Event Service [2] as de facto standard is a rather good mechanism to support decoupled and asynchronous communication. However the Single-Event-Channel-Based Event Service is not fairly good at QoS Control [3-4]. Among many researches, [6] have proposed a fairly good mechanism to support high QoS of CORBA Event Service, in which static and dynamic load balancing mechanism are discussed independently, where fault tolerance discussed in [6] used the same the heartbeat for all nodes, which will work well in a homomorphic environment, but will not be fit for distributed heterogeneous system, especially because of diversity of hardware, software, or critical level of each task. In this paper, we provide a more proper framework for load balancing and fault tolerance, where adaptable load balancing algorithm is utilized to provide high performance of load balancing, node-compliant ping mechanism is adopted for fault tolerance.

## 2 Architecture of the Enhanced Event Service

To gain high QoS, a Hierarchical Master/Slavery Event Channels-Tree framework (HiMSECT) is proposed to support high reliability and availability for CORBA Event Service in push-based information delivery system. In HiMSECT the channels-tree can be viewed as a multi-level structure, which is composed of the master channel, and the slavery ones. Precisely to say, the master channel is just only one channel, which is in charge of receiving data from the source(s) in push mode, while the second slavery channels can be divided into an even smaller master-slavery-channel cell, which is used as both a “master” and “slavery” channel for the communication between its preceding and subsequence channels. Naturally, the last channel should directly connect to the client objects (consumers). The motivation of the HiMSECT framework contributed to the following benefits:

- (1) The availability and the reliability of the system can thus be improved via the hierarchical structure, thus workloads can be distributed fairly among channels;
- (2) The HiMSECT structure achieved a fan-out effect which overcomes the limitation of the number of socket connection for each sender.
- (3) The hierarchical multiple channels-tree framework provides a valid mechanism that support fault tolerance by the failure detection, replication, recovery mechanism.

Formally, a CORBA Event Service is 4-tuple system:  $ES_{basic} = \{S, Ch, C, M\}$ , where  $S$  represents Supplier Objects,  $Ch$  is the Channel object contact suppliers with consumers,  $C$  represents Consumer Objects, and  $M_{Comm}$  represents the communication mode, which consists of 3 kinds of style, that is:

$$M_{Comm} = \{Push, Pull, Push-Pull\}$$

Thus our extended framework can be viewed as the  $ES_{extended}$  expression:

$$ES_{extended} = \{S, W_s, Ch_i, W_c, C, M\}$$

where  $S$ ,  $C$ ,  $M$  have the same meanings as above, but  $W_s$  and  $W_c$  represent the wrapper object of  $S$ , and  $C$  respectively, the most important element  $Ch_i$ , stands the MSECT Event Channels Tree, where

$$Ch_i = \{Ch_{master}, Ch_{Slavery1}, Ch_{Slavery2}, \dots, Ch_{Slaveryn}\}$$

The HiMSECT in push mode is a DAG, where the root node is the master channel Chmaster, and the subnodes are the slavery channels. Either in  $ES_{\text{basic}}$  or  $ES_{\text{extended}}$  Modal, the original service characters are kept both in the above two descriptions. 4 kinds of the communication interaction can be founded as described in CORBA Event Service Specification, that is Push-style, Pull-style, Mixed style, and Multiple Consumers and Multiple Suppliers interaction [1,2].

Theoretically, the HiMSECT framework can doubtless improve the whole system's performance, however, this will be done at the cost of lengthening the propagation delay from the data source to the destinations, then the whole performance will be sacrificed as a result in order to trade for scalability in terms of number of clients supported. Supposing  $C$  represents the cost of creating a channel,  $M$  stands for the communication cost between channels, in a  $n+1$  level channels, if each level has  $m$  layer parallel channels, thus the total cost  $C_{\text{total}}$  can be estimated by the following the expression (  $M_{ij}$  and  $C_{ij}$  represent the  $i$ th level,  $j$ th degree  $M$ -kind and  $C$ -kind cost):

$$C_{\text{total}} = C_{\text{master}} + \sum_{i=1}^m \sum_{j=1}^n (M_{ij} + C_{ij})$$

The goal of the framework includes two aspects, first we want the high performance of QoS, second, we expect the minimum cost in time consumption, such that the goal function can be described as the following formula:

$$\text{Min } C_{\text{total}} = C_{\text{master}} + \sum_{i=1}^m \sum_{j=1}^n (M_{ij} + C_{ij})$$

Subject to:

$$\begin{cases} R_{\text{total}} = \frac{1}{m \times n} \sum_{i=1}^m \sum_{j=0}^n f_r(M_{ij} + C_{ij}) \geq R_0 \\ m \leq M_0 \\ n \leq N_0 \end{cases}$$

where  $f_r(\cdot)$  is the Reliability Decision Function decided by  $(M_{ij} + C_{ij})$ , where  $f_r \in (r_0, 1)$  with  $0 < r_0 < 1$ , thus also  $0 < R_0 < 1$ . Considering the real condition, we should obey the rules that we not only gain more reliability and availability, but constrain the cost in a reasonable and acceptable level, so the layer of the HiMSECT must be constrained in a proper level. For  $\max R_{\text{total}}$  and  $\text{Min } C_{\text{total}}$ , we just keep a 2 level channels: the master/slavery channel ( $N_0=2$ ).

### 3 The 2-Phase Mixed Load Balancing Algorithm

In HiMSECT, static and dynamic load balancing are both used to improve the response time in different stage during run time periods, which is called the 2-Phase Mixed Load Balancing Algorithm(2PMLDAlgorithm). Comparing with [6], we combine the static and dynamic load balancing mechanism together to gain much better

performance. At the first phase, when the system initialed, a basic number of channels created for the coming request on demand, and static load balancing algorithm is used for balancing the channel workload, then with time going, a proper number of channels should be created, to gain real efficiency, the total number of channels must keep in a proper level rather than created without upper bound, thus when a proper number of channels created, for example no more than  $T_{S_0}$ , then dynamic load balancing redistribution strategy is used in the second phase. New request came must first check whether there is a lighter channel among the created ones, if there is, locate the load to the lighter one; else if there is no lighter one and the total channel number is up to the upper bound, then put the new request to the blocked circuit list. After a period of timepiece, the DAEMON will check whether there is a lighter load channel, if there has then select a proper number of blocked request from the blocked list. The 2PMLDAlgorithm is described as following.

**Fig. 1.** The 2PMLDAlgorithm for Load Balancing

```

Input: Subscribe Message Msg with Source info;
Output: IOR of the selected Channel to load the workload;
Status 2PMLDAlgorithm (MsgType Msg, String &IOR)
{
  Submit Msg to MasterServer (Msg);

  for all created channels, calculate avgload  $\bar{L} = \frac{1}{n} \sum_{i=1}^n L_i$ , channel counts  $C_{ch} = \text{Sum}(Ch)$ .

  if ( $\bar{L} < L_{S_0}$  &&  $C_{ch} < C_{ch0}$ ) // Static Load Balancing for Initial Distribution
  {
    Calculate Channel counts  $C_{ch} = \text{Sum}(Ch)$ ;
    if there exists channel with  $L_i < \bar{L}$  //  $i=1,2,\dots,K_0$ 
    {
      Locate the Msg to the Channel with Minload  $L_j$ , where  $L_j = \text{Min}(L_i)$ ,  $L_j < \bar{L}$ .
      IOR=GetCurrentChannelIOR ();
      return S_LOCATE_OK;
    }
    else
    {
      Create a new channel for SubMsg by using interface "EventChannelFactory";
      IOR=GetNewCreatedIOR ();
       $C_{ch} = C_{ch} + 1$ ;
      return S_CREATE_OK;
    }
  }

  else if ( $L_{S_0} < \bar{L} < L_{D_0}$ ) // Adaptive Load Sharing for re-distribution
  {
    put Msg into Waiting Queue WaitQ(Msg);
    for each channel whose  $L_i > L_{S_0}$ , share loads among all channels:
      IOR=GetSharedChannelIOR ();

    until  $S^2 = \frac{1}{n} \sum_{i=1}^n (L_i - \bar{L})^2 < \sigma_0^2$ ;

    get message from WaitQ, and share the load to corresponding channel.
    return S_SHARE_OK;
  }
  return S_UNKNOWN_ERROR;
}

```

In the above description,  $L_{S_0} < \bar{L} < L_{D_0}$  means the average is between the static and dynamic threshold  $L_{S_0}, L_{D_0}$  respectively, while  $C_{ch0}$  stands for the threshold of channel count. The above 2PMLDAlgorithm showed how to locate the new coming request, in fact, the goal of load balancing is to make the current processor work well in a properly fair manner, theoretically, the goal of the load balancing to make the Variance  $S_2$  of  $L_i$  as minimum as possible, here  $\bar{L}$  is the expectation of  $L_i$ . Thus the redistribution can obey the above rules to share the workload among different channels which are already created, with time elapsed the whole load will re-distribute as fairly as possible to each channel, so that the system will work well in a reasonable and acceptable workload level.

#### 4 The Node-Compliant Fault Tolerance Strategy

An important advantage of HiMSECT is that even if one of the subordinate channels fails, there is a good chance the remaining subordinate channels are still functional. However, the consumer(receiver) that connects to the failed channel will be affected. It is necessary to migrate the affected objects to other functional channels so that they can continue to receive data. During the migration the application logic should not be broken, and the interruption of the data service should be minimized in a proper level, this is so-called Fault Tolerance, which is usually based on a timeout mechanism to perform failure detection, node replication and recovery. As the heterogeneity of the hardware or software, setting the same ping cycle for the whole system as described in [6] is not a proper approach, an important reason is that not all tasks have the same request cycle or critical level. So we first classified the tasks into proper critical levels, then setting various ping cycle for each level of task. The Node-Adapted Fault Tolerance Strategy can be described as Table 1:

**Table 1.** The Node-Compliant Fault Tolerance Strategy

The Node-Compliant Fault Tolerance Strategy
<p><b>Step1:</b> For each channel, sending a specified ping message(with different timer cycle) to detect whether the channel works, if there's no response, the channel maybe collapsed;</p> <p><b>Step2:</b> If the channel failed, then the Daemon find the IOR of the collapsed channel, and according to the 2PMLDAlgorithm migrate the current load to a functional channel to continue data dissertation;</p> <p><b>Step3:</b> Try to recover the collapsed channel, if succeed and then put proper load to the recovered channel dynamically;</p> <p><b>Step4:</b> During the running period, if the master Event channel failed, then the daemon process should recover the master channel in the highest priority, for the master channels as the most important server for all other slavery channels.</p>

The daemon in HiMSECT is a global self-governed process controlled by the system, but not by the master event channel. So either the slavery channel or the master channel failed, the daemon can monitor it and recover it timely.

5    Performance Evaluation

A light prototype of HiMSCET has been implemented using Microsoft Visual C++ 6.0 and VisiBroker for C++ to test the 2PMLDAlgorithm. 6 different requests various from 10 to 75 are used to evaluate the load balancing time, for each specific request 5 groups of experiment results are recorded. Where we just used master/slavery, the parallel channel is 4, that is in (\*), $N_0=2, M_0=4$ . The hardware and software environment: Pentium®III 733MHz-128M LEGEND PC. Table 3 gives the time elapsed for load balancing in which the average time are all lower than those in [6]).

**Table 2.** Time Elapsed for Load Balancing in Various Request Environment

ClientRequestNumber	10	25	40	45	60	75
Group 1	765	1012	1302	1648	2276	2708
Group 2	804	1197	1348	1599	2127	2892
Group 3	828	1134	1409	1607	2323	2787
Group 4	831	1203	1482	1578	2296	2809
Group 5	798	1265	1335	1694	2292	2845
AVG	805	1162	1375	1625	2262	2808

6    Conclusion

In this paper, we contributed to the adaptive load balancing algorithm for optimal workload distribution and node-compliant fault tolerance mechanism for failure detection, node recovery for the event-driven communication. In which the adaptive two phase load balancing mechanism that considering static and dynamic load distribution together worked well in smart mode to adapt to various client nodes which optimize the system resources in a reasonable level. And the node-compliant tolerance strategy is efficient for the adoption of an adaptive heartbeat (ping). Large amount of simulations showed that the proposed algorithms is reliable and available.

References

1. Tanenbaum, A.S. Distributed Operating System, Englewood, NJ: Printice Hall. (1994)
2. OMG: CORBA Event Service Specification 1.0, <http://www.omg.org>.
3. Schmidt, D. C.: Object Interconnections: Overcoming Drawbacks with the OMG Events Service. SIGS, Vol. 9, (1997)
4. Antonio Carzaniga,et al.: Achieving scalability and expressiveness in an internet-scale event notification Service. Symposium on Principles of Distributed Computing (2000):219-227
5. Ghosh, Bhaskar; Muthukrishnan, S.: Dynamic load balancing by random matchings. Journal of Computer and System Sciences, Vol.53(3),(1996):357-370
6. Kei Shiu Ho,et al.: An extended CORBA event service with support for load balancing and fault-tolerance. Proceedings of Distributed Objects and Applications (2000):49 -58
7. Dasgupta, Pallab; Majumder, A. K.; Bhattacharya, P.: V\_THR: An adaptive load balancing algorithm. Journal of Parallel and Distributed Computing, Vol.42(2), (1997):101-108
8. Prasetya, I.S.W.B.; Swierstra, S.D.: Factorizing fault tolerance. Theoretical Computer Science, Vol.290(2),(2003):1201-1222

# Performance Optimization of Fractal Dimension Based Feature Selection Algorithm\*

Yubin Bao, Ge Yu, Huanliang Sun, and Daling Wang

School of Information Science & Engineering,  
Northeastern University, Shenyang 110004, P.R.China  
{baoyb, yuge}@mail.neu.edu.cn

**Abstract.** Feature selection is a key issue in the advanced application fields like data mining, multi-dimensional statistical analysis, multimedia index and document classification. It is a novel method to exploit fractal dimension to reduce dimension of feature spaces. The most famous one is the fractal dimension based feature selection algorithm *FDR* proposed by Traina Jr et al. This paper proposes an optimized algorithm, *OptFDR*, which scans the dataset only once and avoids the efficiency problems of multiple scanning large dataset in the algorithm *FDR*. The performance experiments are made for evaluating *OptFDR* algorithm using real-world image feature dataset and synthetic dataset with fractal characteristics. The experimental results show that *OptFDR* algorithm outperforms *FDR* algorithm.

## 1 Introduction

*Feature selection* means to select a subset of  $m$  features from a dataset of  $N$  features ( $m < N$ ), where the  $m$  features can also discriminate each data object in the dataset as the  $N$  features do. Feature selection is a key issue in the advanced application fields such as data mining, multi-dimensional statistical analysis, multimedia index and document classification. In these applications, usually large datasets are involved, which comprise large number of attributes (features) and records. In practice, these features are not independent but related each other [1]. By feature selection, the irrelevant features to classification attributes in supervised learning or redundancy features are reduced so that the effectiveness of analysis tasks can be increased and the accuracy of prediction can be improved [2,3]. There are many feature selection algorithms using many kinds of techniques [3,4,5]. Traina Jr et al [6] proposed a method, *FDR*, to use fractal dimension for feature selection to support on-line processing. Moreover, the method doesn't need to do feature transformation or rotation so that the resulting features are easy to understand. The method is good at to estimate the number of features to be kept in the dataset. But the algorithm has the efficiency shortcoming since it needs scanning dataset in multiple times. In this paper, we de-

---

\* Supported by the National Natural Science Foundation of China under Grant No.60173051, the Teaching and Research Award Program for Outstanding Young Teachers in Higher Education Institutions of the Ministry of Education, China.



signed a new optimized algorithm, *OptFDR*, in which a fractal tree is used to computing fractal dimension and the fractal dimension after reducing a feature can be computed just through adjusting the fractal tree. Therefore, the *OptFDR* algorithm only needs to scan the dataset once to determine which features are to be reduced. The experimental results show that *OptFDR* algorithm outperforms *FDR*.

The paper is organized as follows. Section 2 introduces the basic concepts of fractal and fractal dimension. Section 3 discusses the method of fractal dimension computation and feature selection. Section 4 presents the optimized feature selection algorithm *OptFDR*. Section 5 gives performance comparisons between the new algorithm *OptFDR* and the old algorithm *FDR*. Finally, section 6 concludes the paper.

## 2 Fractal and Fractal Dimension

If a dataset has self-similarity in all observation metrics, that is, the partial distribution of the dataset has the similar structure or features as its whole distribution, then the dataset is said as *fractal* [6]. Next, we give the concepts related with dataset dimensionality.

The dimensionality of the Euclid space where the data points of a dataset exist is called as the *embedding dimension* of the dataset, that is, the number of the features of the dataset. The dimension of a real space object is less than that of its Euclid space. For example, a line whether in a 2-dimensional space or in a 3-dimensional space is actually 1-dimensional. The actual dimensionality of the space objects that a dataset represents is called as the *Intrinsic Dimension* of the dataset; no matter what space dimensionality it exists. Suppose a dataset has the embedding dimension  $E$ , and the intrinsic dimension  $ID$ , then  $ID \leq E$ . The *fractal dimension* of a dataset represents the native property of the dataset. For a dataset  $d$  with embedding dimension  $E$ , we embed  $d$  into an  $E$ -dimension grid in which each cell has the side length  $r$ ,  $r \in (r_1, r_2)$ , and  $(r_1, r_2)$  is the varying range of the side length measure with the fractal property of the dataset. Thus, the number of the data points that falls in the  $i$ th cell, denoted as  $C_i$  can be calculated. The fractal dimension  $D_q$  of the dataset is defined by formula 1.

$$D_q = \frac{1}{q-1} \frac{\partial \log \sum_i C_i^q}{\partial \log r}, \quad r \in (r_1, r_2) \quad (1)$$

In formula 1, when  $q=2$ ,  $D_2$  is called as *correlation fractal dimension*, which describes the probability of that two randomly selected data points fall within a range. The change of  $D_2$  means the change of the distribution of data points in the dataset. In the following sections, we use the *correlation fractal dimension* as the measure of the dataset intrinsic dimension.

## 3 Fractal Dimension Calculation and Feature Selection

A *correlation fractal dimension* can be estimated by using Box-counting dimension. The calculation method is presented in [6]. In that method, a *fractal tree* is to be built

to record the number of data points that falls into each cell, with respect to different cell side length level  $r$ . The tree node of each cell contains two parts of information: the number of data points fallen in, denoted as  $C$ , and the pointer to the nodes of the sub-cells in the next dividing level. Except for the root node, each node contains the information of  $2^E$  cells. Normally, a node contains many cells with  $C$  equal to 0, so a list can be used to implement each node such that only the cell that contains data points are recorded in the node. By this way, two pieces of information are added in each cell: the internal number  $I$  (the cell number within a node), and pointer  $PN$  to the next cells within the same node (see Fig.1). In Fig.1, the root pointer *Root* points to the level corresponding to  $r=1/2$ , the second level corresponds to  $r=1/4$ , and et al.

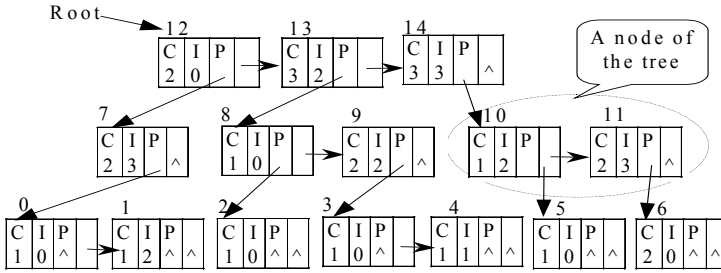


Fig. 1. A 2-dimensional fractal tree.

The computation of the fractal dimension of a dataset only needs to scan dataset once. Based on the algorithm, Traina Jr et al [6] proposed a backward feature reduction algorithm *FDR*. The bottleneck of the performance of *FDR* is to call the algorithm of computing fractal dimension repeatedly to compute the *partial fractal dimension* ( $pdf$ ) of the different combinations of the candidate features. So, we optimize the algorithm as follow.

## 4 Optimized Fractal Feature Selection

In *FDR* algorithm, to reduce a feature from  $E$  features,  $E$  times scanning on the dataset is needed; then  $(K*(2E-K+1))/2$  times scanning is needed in order to reduce  $K(K < E)$  features. For a large dataset with large number of features to be reduced, the I/O overhead will be heavy. Let  $E$  represent the number of features of a dataset, and Let  $T(E)$  represent the fractal tree with  $E$  features. We found that the scanning on the dataset is not necessary to build the fractal tree  $T(E-1)$  after a feature is reduced from the fractal tree  $T(E)$ . It just need to adjust the fractal tree  $T(E)$  by using a simple algorithm.

### (1) Adjusting Fractal Tree

Without lose of generality, suppose  $E$  features are arranged in a fixed sequence, and the two cells  $gi$  and  $gj$  in a node of  $T(E)$  have sequence number  $gi_E$  and  $gj_E$ , respectively. Let  $gi_E$  and  $gj_E$  represent in binary mode as follows:  $gi_E = (xi_{E-1}, xi_{E-2}, \dots, xi_{i+1},$

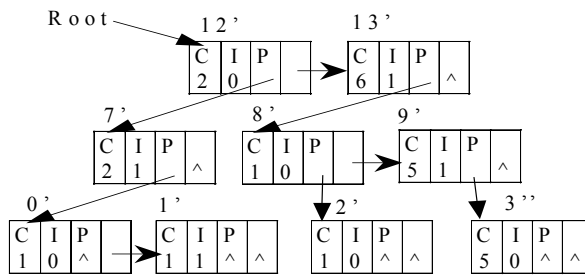
$x_{i_1}, x_{i_{i-1}}, \dots, x_{i_1}, x_{i_0})_2, g_{j_E} = (x_{j_{E-1}}, x_{j_{E-2}}, \dots, x_{j_{i+1}}, x_{j_i}, x_{j_{i-1}}, \dots, x_{j_1}, x_{j_0})_2$ . Where,  $x_{i_p}, x_{j_i}$  represent the values of the  $i$ th dimension and the value is 0 or 1. If the  $i$ th dimension is reduced from the  $E$  dimensional space, it is equivalent to project the  $E$  dimensional space onto the  $E-1$  dimensional spaces. The cell sequence number after projecting as follows:  $g_{i_{E-1}} = (x_{i_{E-1}}, x_{i_{E-2}}, \dots, x_{i_{i+1}}, x_{i_{i-1}}, \dots, x_{i_1}, x_{i_0})_2, g_{j_{E-1}} = (x_{j_{E-1}}, x_{j_{E-2}}, \dots, x_{j_{i+1}}, x_{j_{i-1}}, \dots, x_{j_1}, x_{j_0})_2$ . The condition to merge cells  $g_{i_E}$  and  $g_{j_E}$  after reducing  $i$ th dimension is:  $g_{i_{E-1}} = g_{j_{E-1}}$ , and the new sequence number is  $g_{i_{E-1}}$  or  $g_{j_{E-1}}$ .

Based on the above discussion, the performance of the fractal feature selection algorithm *FDR* can be optimized. The main idea is that the computation of *partial fractal dimension* with  $n$  features is unnecessary to scan dataset, but by adjusting the *fractal tree* with  $n+1$  features. The *OptFDR* algorithm is designed according to this idea.

To adjust the fractal tree, the major problem is how to find the cell pairs to be merged in each node of the  $E$  dimensional fractal tree. To merge the cells means to add their value  $C$  as the value  $C$  of the new cell, and combine the sub-trees rooted with them. Sub-tree combination means to combine the cells with the same sequence number (after adjusting) and within the same level, keep or immigrate the cells with difference sequence number.

For example, the adjusting process from Fig.1 to Fig.2 after deleting the 0<sup>th</sup> dimension is as follows: The cell 3 and 4 in Fig.1 are combined into a cell 3' with  $I=0$ ; the cell 10 and 11 in Fig.3 are combined into cell 10' with  $I=1$ , then it makes cell 5 and 6 in Fig.3 be combined into cell 5'; the value  $I$  of cell 9 is adjusted to 1; the cell 13 and 14 are combined into cell 13', and their children are also combined if the sequence number is the same after adjustment (i.e. cell 9 and 10' are combined into 9', and cell 3' and 5' are combined into 3'').

The fractal tree adjustment algorithm, *AdjustFTree*, can be straightforward designed according to the above discussed. Because of the space limit, it is omitted.



**Fig. 2.** 1-Dimensional fractal tree after deleting the 0<sup>th</sup> Dimension of the tree in Fig. 1.

## (2) Optimize Fractal Feature Selection

By calling algorithm *AdjustFTree*, the fractal tree with  $E-1$  features can be obtained from the one with  $E$  features. Of course,  $E-1$  features are the sub-set of  $E$  features. Next, *OptFDR*, the optimized fractal feature selection algorithm based on algorithm

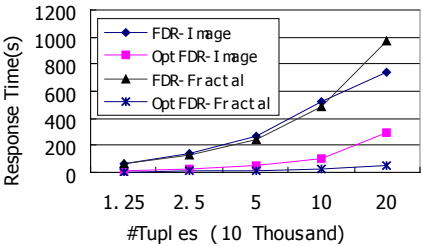
*AdjustFTree* can be described as algorithm 1. The *OptFDR* algorithm optimizes the computation of *pdf* when removing a feature. The algorithm is omitted for space limit.

Since we can calculate the fractal dimension of the  $E-1$  dimension dataset from the derived fractal tree of the  $E$  dimension fractal tree, the partial fractal dimension *pdf* can be got without needing to scan dataset as in *FDR* algorithm. In this approach, no matter how big  $E$  is, only 3 fractal trees are stored. The space complexity of *OptFDR* is  $O(3*N*H)$ , while that of *FDR* is  $O(N*H)$ , where  $N$  is the number of tuples, and  $H$  is the height of the fractal tree, or the level of data space dividing. So the storage increases two times than *FDR* algorithm. On the aspect of I/O, *FDR* algorithm needs to scan dataset  $(K*(2E-K+1))/2$  times when reducing  $K(K<E)$  features, where  $E$  is the total number of features. However, *OptFDR* scans dataset only once for reducing features. Furthermore, the performance comparisons will be discussed at section 5.

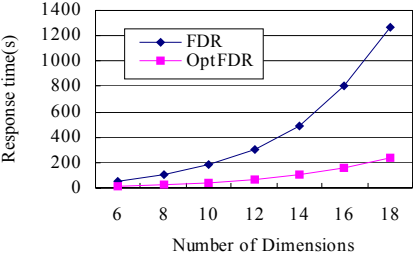
### 5 Performance Comparisons and Analysis

Two typical experiments were made to test the performance of *OptFDR* algorithm and compare it with *FDR* algorithm. The hardware environment includes Intel Pentium IV 1.0GMHz CPU, 256MB RAM, 20GB hard disk, and the software environment includes Windows 2000, VC++, and SQL Server.

The first experiment is about the relationship between the number of tuples and time overhead of the algorithm. Two testing dataset are selected. The first dataset is an image feature dataset with 32 Fourier transformation coefficients [7], and 20 features are selected to generate the testing dataset. The second dataset is a synthetic dataset with 8 features [8]. Among them the features  $a$  and  $b$  are fractal data, and the rest features are the transformation from features  $a$  and  $b$ .



**Fig. 3.** The performance comparisons. “-Image” represents using image data, “-Fractal” represents using fractal data.



**Fig. 4.** The performance comparison.

Fig.3 shows that the response time of two datasets is different very much. The reason is that the data distribution of the fractal dataset has obvious fractal properties. As the finer dividing the space, the number of the cell that contains data points doesn't increase obviously. Thus, the time overhead doesn't change obviously as the number of tuples increases. That shows the advantage of the optimized algorithm. While the image-feature data doesn't have obvious fractal properties, the number of the cell containing data points also increases obviously as the number of tuples increases.

Thus, the time for adjust the fractal tree also increases obviously. But comparing with *FDR*, *OptFDR* still has good performance.

Fig. 4 shows the relationship between dimensionality and the time overhead of the two algorithms. The testing dataset is the image-feature dataset with 20,000 tuples. The processing is to reduce last two dimensions. The response time of *OptFDR* with respect to dimensionality is much lower than that of *FDR*. It is much better than *FDR*.

The experimental results show that the performance of *OptFDR* outperforms *FDR*'s.

## 6 Conclusions and Future Work

Through calculating the fractal dimension of a dataset, the lower limit of the number of the features to be kept can be determined. An optimized fractal dimension based feature selection algorithm is proposed, which can complete the feature selection process through scanning the dataset once except for preprocessing.

The fractal dimension based feature selection algorithm is an unsupervised feature selection algorithm, that is, it is can be used for the dataset without classification information. This approach can give the optimal feature subset under fractal dimension measure, and the feature ordering according to their importance.

Through the experimental results show that fractal dimension based feature selection algorithm can remove redundancy features, and fractal dimension is a good measure for feature subset. So, we will combine the fractal dimension of a dataset with some efficient feature selection algorithms to get some more efficient algorithms.

## References

1. B. U. Pagel, F. Korn, C. Faloutsos, Deflating the Dimensionality Curse Using Multiple Fractal Dimensions, ICDE'2000, San Diego, CA - USA, 2000:589-598
2. Talavera, L. Feature selection as a preprocessing step for hierarchical clustering, Proc. of the 16th Intl. Conf. on Machine Learning(ICML'99), 1999:389-397.
3. Arnaud Robert, Emmanuel Stocker, Abdel Ennaji, Yves Lecourtier. Classification and Feature Selection by a Self-Organizing Neural Network. Proc. of Intl. Work-Conf. On Artificial and Neural Networks, IWANN (1), Alicante, Spain, 1999: 651-660.
4. Franz Pernkopf, Paul O'Leary. Feature Selection for Classification Using Genetic Algorithms with a Novel Encoding. Proc. of CAIP'2001, Warschau, 2001: 161-168.
5. Boussouf M, Quafafou M. Scalable feature selection using rough set theory. Ziarko W, eds. Proc of 2nd Int. Conf. on Rough Sets and Current Trends in Computing Banff, Canada: Springer, 2000:131-138
6. Caetano Traina Jr., Agma Traina, et al. Fast feature selection using fractal dimension, Proc. of the 15th Brazilian Symposium on Databases, Paraila, Brazil, 2000(SDDB'2000).
7. Berchtold S. Founew.normiert.gz image dataset. Available at <http://www.stb-ag.com/~berchtold/papers/founew.normiert.gz> 2002-09-17
8. Mark Finlay, Keith A. Blanton. Real world Fractal, M&T Publishing Press, USA, 1995.

# Graph-Based Cyclic Multi-join Query Optimization Algorithm

Hong Yu<sup>1,2</sup>, Xiu-Kun Wang<sup>1</sup>, and Jian-Ying Zhang<sup>1</sup>

<sup>1</sup> School of Elect. and Info. Eng., Dalian Univ. of Tech., Dalian 116024, China

<sup>2</sup> School of Information Eng., Dalian Univ. of Fish., Dalian 116024, China

**Abstract.** Although many Multi-Join query optimizations algorithms have been proposed, there is a lack of cyclic multi-join query optimization algorithms. In this paper, a graph-based cyclic multi-join query optimization algorithm-Improved Minimum Result Relation-Minimum Weight First (Improved-MR-MWF) was proposed. The time complexity is  $O(mn)$ . This algorithm, considering the influences among the join operations concerning same relation, chooses the join that the result relation is minimum as the top-priority operation. The minimum weight join among the top-priority operations was selected. The experimental results indicated that the cost and the query response time of the new algorithm were less than that of others.

## 1 Introduction

Multi-Join is one of the hot topics in query optimizations. Various methods were proposed to improve join's efficiency [1–5]. A graph-based method was proposed by Chiang Lee[1]. In this model, the graph included all possible execution plans. Each spanning tree of the graph was an execution plan. A Maximum Value Priority algorithm (MVP) was devised to find a near optimal execution plan. The time complexity of MVP is  $O(n^2)$ . Another graph model was proposed by Ming-Syan Chen[2]. The steps of multi-join query optimization were demonstrated and four heuristic algorithms were proposed. The comparison among four algorithms indicated that  $G_{MC}$  and  $G_{MR}$  surpassed the other two algorithms. The time complexities of  $G_{MC}$  and  $G_{MR}$  are all  $O(n^2)$ . Although all these methods can improve the efficiency of the join operations in some extent, some of them didn't consider the mutual influence of the join operations concerning same relation and the influence of the weight of the result relation on the succeeded join. Still, there is a lack of cyclic multi-join query optimization algorithms. In fact, there are a lot of multi-join query is cyclic, for example,

Select \* from R1,R2,R3 where R1.A=R2.A and R2.B=R3.B and R3.C=R1.C

In order to solve all sorts of multi-join query, we devised a new graph model on the basis of reference [2] and proposed an Improved Minimum Result Relation-Minimum Weight First ( Improved-MR-MWF) algorithm to search for a optimal execution plan.

After researching on the multi-join query optimization, we found that a multi-join query execution plan must take the following three factors into considera-

tion: (1) the mutual influence of the join operation, especially the mutual influence of the join operation concerning the same relation. (2) The join operation enlarges or reduces the result relation. We took this factor as weight. If the join enlarges the original relations, the enlargement would be spread to the succeeded joins. The cost of this execution plan would be expensive. (3) the merge of the joins with cyclic.

According to the researching, we found that the above algorithms have some limitations. The MVP algorithm didn't take the influence of the cardinality of the result relation into consideration.  $G_{MR}$  neglected the weight of the join operation. All of the above algorithms didn't consider the cyclic join operation. We proposed graph-based cyclic join-operation model, on the basis of this model, we proposed an Improved Minimum Result Relation Minimum Weight Algorithm. The rest of the paper is organized as follows: In section 2, the definition of the graph model was presented, the vertex, the edge, the weight of vertex, the weights of edge are also defined. In section 3, the algorithm is described. In section 4, the experiment result is given and the time complexity is analyzed. In section 5, the conclusion and future work is presented.

## 2 Definition of Graph Model

In our model, the relationship among the join operations that concerned the same relation must be embodied. Some relations only take part in one join operation. The result relation of the join operation can affect only this relation. The influence cannot spread to other join. Other relations take part in more than one join. The result relation of the joins can affect the succeeded join. We must know the number of the join that a relation takes part in. We improved the graph model in [2]. Each vertex has a weight—degree. The degree of the vertex denotes the number of the join that the relation takes part in. edges denote joins. We must know about the cardinality of the result relation and the influence factor of the join. In addition, we must know the join attributes of each join operation. We set a weight group  $w(w_1, w_2, w_3, w_4, w_5, w_6)$ , where  $w_1$  is Join Selectivity Factor(JSF). The definition of JSF refer to reference [1]:

$$JSF_i = \frac{|R_p JOIN_i R_q|}{|R_p| * |R_q|} \quad (1)$$

Where  $|X|$  means the cardinality of relation X

$w_2$ : the cardinality of the result relation of  $R_p$  join  $R_q$ .

$w_3$ : is the influence of  $R_p$  join  $R_q$  on  $R_p$ , after  $R_p JOIN_i R_q$ , all join operation that  $R_p$  takes part in would replace  $R_p$  with the result relation of  $R_p JOIN_i R_q$ ,  $R_p JOIN_i R_q$  would affect the other join that  $R_p$  takes part in. the influence factor is as follows:

$$w_3 = \frac{|R_p JOIN_i R_q|}{|R_p|} = \frac{|R_p JOIN_i R_q| * |R_q|}{|R_p| * |R_q|} = w_1 * |R_q| \quad (2)$$

$w_4$  is the influence on  $R_q$  of  $R_p$  join  $R_q$ ., For the same reason,  $w_4 = w_1 * |R_p|$

There are two relations to take part in the join operation. We must consider the overall influence on the two relations.

$w5$  is the overall influence on the two relations.  $w5 = w3 * w4$

$w6$  is the set of attribute of the join operation.

The definition of the Graph:

**Definition 1.** *Multi-Join Query Graph  $G(V, E)$  is a weighted graph, where, the set of the vertices is  $G(V) = \{R1, R2, \dots, Rm\}$ , which is the set of the relations that take part in the joins, each vertex has a weight-degree, which means the number of the join that the relation takes part in. The set of the edges is  $G(E) = \{join_1, join_2, \dots, join_n\}$ , which is the set of joins. Where  $join_i = R_p join_i R_q, R_p, R_q \in G(V)$ , each edge— $join_i$  has a weight set:  $\{w1, w2, w3, w4, w5, w6\}$*

$join_i.w1 = JSF_i$ , means the Join Selectivity Factor  $join_i.w2 = join_i.w1 * |R_p| * |R_q|$  the cardinality of the result relations.

$join_i.w3 = join_i.w1 * |R_q|$  the influence on the relation  $R_p$  of the join

$join_i.w4 = join_i.w1 * |R_p|$  the influence on the relation  $R_q$  of the join

$join_i.w5 = join_i.w1 * join_i.w2$  the overall influence on the two join relations

$w6$  the set of attribute of the join operation.

Where  $|X|$  means the cardinality of relation  $X$

### 3 Graph-Based Multi-join Query Optimization Algorithm

#### 3.1 The Algorithms

In algorithm Improved-MR-MWF, we must get the best join of each relation. Firstly, we give algorithm BEST-JOIN to get the best join associated with each relation. Then we give the Improved-MR-MWF algorithm. The algorithms are as follows:

// get the minimum result relation join of the multi-joins that relation  $R$  take part in

*Algorithm best-join( $R$ )*

1.  $join = \epsilon$  //join means a join operation  
size =  $\infty$  //size means the size of the result relation
2. for all  $R \Join_i R' \in E$  do  
if  $Join_i.w2 < size$  then  $join = Join_i$  :  $size = Join_i.w2$   
end for
3. return join
4. end best-join

//Improved Minimum Result Relation Minimum Weight First Algorithm  
Algorithm Improved-MR-MWF()

INPUT: A multi-join operation query graph  $G(V, E)$

OUTPUT: a optimal multi-join execution plan  $L$  Symbols:

$k$ : the maximum suffix of the relation in  $G(V)$ , initial  $k = m$

$L$ : execution plan sequence, initial is empty.

//to get the best join of all relations whose degree is  $> 1$



1. for all  $R_i \in G(V)$  do
  - if  $R_i.degree > 1$  then  $R_i.join = \text{best-join}(R_i)$
  - End for
2.  $w = \infty$  //  $w$  means the weight of the best join  
 $join = \varepsilon$  // variant join means the best join  
 // to get the minimum weight join of the best join  
 for all  $R_i \in G(V)$  do if  $R_i.degree \leq 1$  then  
 if  $R_i.join.w \leq w$  then  $w = R_i.join.w$  :  $join = R_i.join$   
 end if  
 end for
3. if  $join = \varepsilon$  then goto(8)
4. do the join operation that the variant join represents,  $R_p \text{ join}_i R_q \in G(E)$ ,  
 where  $R_p, R_q \in G(V)$ .
5. for all  $R_i \in G(V)$   
 if  $(R_i, R_p) \in G(E)$  and  $(R_i, R_q) \in G(E)$  then  
 if  $(R_i, R_p).w \neq (R_i, R_q).w$  then  
 $(R_i, R_p).w = (R_i, R_p).w \cup (R_i, R_q).w$   
 $(R_i, R_p).w1 = (R_i, R_p).w1 * (R_i, R_q).w1 : G(E) = G(E) - (R_i, R_q)$   
 End if  
 End if  
 End for
6. // to get the result relation  $R_k$  of the join operation  
 $k = k + 1$ :  $R_k = R_p \text{ join}_i R_q : G(V) = G(V) \cup \{R_k\}$   
 $|R_k| = join_i.w2 : R_k.degree = R_p.degree + R_q.degree - 2$   
 // replace  $R_p$  and  $R_q$  with  $R_k$  in new  $G$ .  
 For all  $R_a \text{ join}_j R' \in G(E) (R_a = R_p \text{ or } R_q)$   
 $G(E) = G(E) \cup R_k \text{ join}_j R'$   
 If  $|R_k| < |R_a|$  then  
 $join_j.w2 = join_j.w1 * |R'| * |R_k|$   
 $join_j.w3 = join_j.w1 * |R'|$   
 $join_j.w4 = join_j.w1 * |R_k|$   
 $join_j.w2 = join_j.w3 * join_j.w4$   
 end if  
 $G(E) = G(E) - \{R_a \text{ join}_j R'\}$   
 End for  
 // delete the finished join and the relative relations  
 $G(E) = G(E) - \{join_i\} : G(V) = G(V) - \{R_p, R_q\} : L = L \cup \{join_i\}$
7. for  $R_k$  and relative relations run best-join, goto(2)
8. output  $L$ , Stop

### 3.2 Time Complexity

The time complexity of Improved MR-MWF is analyzed in the following:

- STEP 1. for each relation to run algorithm best-join(), time complexity is  $O(m^2)$ , where  $m$  is the number of the vertices of graph  $G$ .
- STEP 2. to make sure the minimum weight join, in the worst case, the time complexity is  $O(m)$
- STEP 3. takes a constant time  $c$
- STEP 4. takes a constant time  $c$
- STEP 5. to merge the repetition edge, the time complexity is  $O(m)$
- STEP 6. to modify the graph  $G$ , in the worst case, the time complexity is  $O(m)$
- STEP 7. in the worst case, the time complexity is  $O(m)$

The time complexity of step 2 to step 7:  $O(m) + c + c + O(m) + O(m) + O(m) = O(m)$

There are  $n$  repetitions from step 2 to step 7, the time complexity is  $O(mn)$ , where  $n$  is the number of edges in graph  $G$ .

The time complexity of the algorithms is  $O(m^2) + O(mn)$ , for  $m - 1 \leq n < m^2/2$ , the time complexity of the algorithm is  $O(mn)$

## 4 Experiment Result Analysis

The algorithm was used in a flood-protected system. The experiment environment is SUN workstation. We select 15 queries randomly. Some of them are cyclic join and the others aren't cyclic join. We compare the query cost and the query response time with other algorithms the result are showed in Fig. 1 and Fig. 2

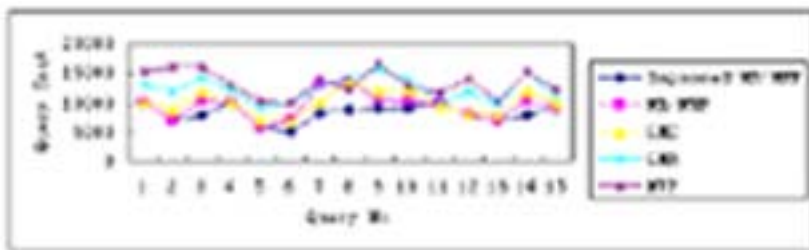


Fig. 1. The comparison of the query cost

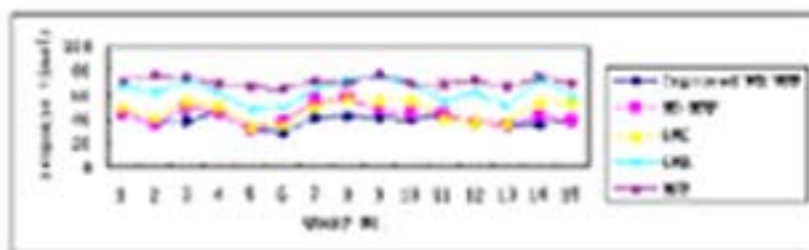


Fig. 2. The comparison of the response time

respectively. From the result, we can see that the cost of Improved-MR-MWF algorithm is lower than other algorithms and the response time is less than other algorithms as a whole. We can conclude that the Improved-MR-MWF algorithm is better.

## 5 Conclusion and Future Work

After we analyzed several past multi-join query optimization algorithms, we found that each algorithm has its own limitation. a algorithm with time complexity of  $O(mn)$  was proposed, the experiment result indicated that the cost of the query is lower than other algorithms and the response time of this algorithm is less than other algorithm.

The algorithm is applicable for single processor. If there are multiple processors, the algorithm must be further improvement. The next work is to research the improved algorithm in multiple processors and the processor allocation algorithm.

## References

1. Chiang Lee, Chi-Sheng Shih, Yaw-Huei Chen.: Optimizing Large Join Queries Using A Graph-Based Approach. IEEE Trans on Knowledge and Data Engineering **Vol.13 No. 2** (2001) 298–315
2. Ming-Syan Chen, PhilipS.Yu, Kun-Lung Wu. :Optimization of Parallel Execution for Multi-Join Queries. IEEE Trans on Knowledge and Data Engineering, **Vol.8 No.3** (1996) 416–428
3. Lee, Z. A. Chang. : Utilizing Page-level Join Index for Optimization in Parallel Join Exec-ution. IEEE Trans on Knowledge and Data Engineering, **vol.7,no.6** (1995) 900–914.
4. K.A. Hua, C. Lee, C.M. Hua. : Dynamic Load Balancing in Multi computer Database System Using Partition Tuning. IEEE Trans on Knowledge and Data Engineering, **Vol.7 No.6** (1995)968–983.
5. CAO Yang, FANG Qiang, WANG Guo-ren, YU Ge. :Parallel Query Optimization Tech-niques for Multi-join Expressions Based on Genetic Algorithm. Journal of software, **Vol.13, No.2** (2002) 250–257

# Author Index

- Aghbari, Zaher Al 229  
Aldana-Montes, José Francisco 249  
An, Jiyuan 65  
Au, Pui-On 127, 176
- Baeza-Yates, Ricardo 315  
Baier, Jorge 315  
Baik, Doo-Kwon 239, 684  
Bang, Nanhyo 708  
Bao, Jun-Peng 640  
Bao, Yubin 600, 739
- Canós, Jose H. 449  
Chang, Elizabeth 1  
Chen, An-long 55  
Chen, Ding-Yi 499  
Chen, Guoning 658  
Chen, Jixiong 97  
Chen, Min 539  
Chen, Wenliang 646  
Chen, Yen-Liang 569, 579  
Chen, Yi-Ping Phoebe 65  
Chen, Zhi-ping 678  
Cheng, Leung 127  
Choi, Chi-Hon 147  
Chung, Ki-Dong 117
- Dai, Yiqi 268  
Deng, Bing 720  
Deng, Shengchun 589, 726  
Dillon, Tharam S. 1
- Ezeife, Christie I. 539
- Fan, Zhiping 720  
Feng, Boqin 76, 733  
Fu, Haohuan 176  
Fu, Lixin 336  
Fu, Tao 419
- Gao, Jun 389, 399, 559  
Ge, Yan-Feng 208  
Guan, Jianbo 664  
Guo, Longjiang 24
- He, Zengyou 589, 726
- He, Zhenying 378  
Hsu, Wynne 528  
Huang, Joshua Zhexue 489, 549, 589, 726  
Huang, Shangteng 628  
Huang, Shen 208  
Hussain, Farookh 1
- Jaén, Javier 449  
Jeong, Dongwon 239, 684  
Jia, Weijia 127, 176  
Jia, Yan 664  
Jiang, Changjun 280  
Jiang, Zhou 107  
Jin, Cheqing 549  
Jin, Jesse J. 429  
Jung, Eui-Hyun 325
- Kao, Hung-Pin 579  
Keogh, Eamonn 65  
Kim, Dong-kyoo 87  
Kim, Han-joon 519  
Kim, Je-uk 519  
Kim, Wonil 87  
Kim, Young-Gab 239, 684  
Ko, Ming-Tat 579  
Küngas, Peep 458
- Le, Jia-Jin 45  
Lee, Gunhee 87  
Lee, Mong Li 528  
Lee, Ook 610  
Li, Chuan 55  
Li, Guohui 97  
Li, Jianqing 696  
Li, Jianzhong 24, 378, 634, 652, 690, 702  
Li, Jieyu 303  
Li, Jinbao 652  
Li, Rui 478  
Li, Shijun 714  
Li, Taoshen 658  
Li, Xue 499  
Li, Yin 167  
Lim, Chee Chern 429  
Lin, Xuemin 34, 218  
Lin, Ya-ping 678

- Lin, ZuoQuan 696  
 Ling, Tok Wang 714  
 Liu, Baoliang 690, 702  
 Liu, Chengfei 346  
 Liu, Da-you 137  
 Liu, Hai-Yan 640  
 Liu, Huan 622  
 Liu, Jian-Wei 45  
 Liu, Jixue 346  
 Liu, Mengchi 714  
 Liu, Shengping 696  
 Liu, Wenyin 280  
 Liu, Xiao-Dong 640  
 Liu, Xinyu 720  
 Liu, Yin 280  
 Liu, Yunfeng 399, 559  
 Long, Zhiyi 468  
 Lu, Hai-Xin 218  
 Lu, Hongjun 147  
 Lu, Jing-Tin 569  
 Lu, Ren 528  
 Luo, Jizhou 634  
 Luo, Yi 218, 468  
 Luo, Yingwei 186
- Ma, Fanyuan 167, 489, 672  
 Ma, Xiuli 389  
 Ma, Zhaofeng 76, 733  
 Makinouchi, Akifumi 229  
 Mandvikar, Amit 622  
 Matskin, Mihhail 458  
 Meng, Weiyi 197  
 Meng, Xiaofeng 409  
 Mody, Jigar 622
- Navarro, Elena 449  
 Navas-Delgado, Ismael 249  
 Nie, Tiezheng 478  
 Nussbaum, Miguel 315
- Pan, Leyun 672  
 Park, Si-Yong 117  
 Peng, Dunlu 439  
 Peng, Ya 678  
 Peng, Zhiyong 468, 714
- Qi, Wenwen 368
- Rao, Jinghai 458  
 Ren, Yi 664
- Roldán-García, María del Mar 249  
 Rong, Hongqiang 489
- Saxton, Lawrence V. 357  
 Shan, Zhe 468  
 Shen, Derong 478  
 Shen, Jun-Yi 640  
 Shi, Fei 259  
 Shi, Shengfei 652  
 Shim, Kyuseok 23  
 Song, Hui 672  
 Sun, Huanliang 600, 739  
 Sun, Shouqian 616
- Tang, Chang-jie 55  
 Tang, Haidong 720  
 Tang, Shiwei 389, 399, 559  
 Tang, Xiqun 357  
 Tang, Yongchuan 616
- Um, Kyhyun 708
- Vincent, Millist W. 346
- Wang, Chaokun 652  
 Wang, Daling 600, 739  
 Wang, Haixun 409  
 Wang, Hongya 97  
 Wang, Hongzhi 378, 634  
 Wang, Huayong 268  
 Wang, Shan 409  
 Wang, Sheng-sheng 137  
 Wang, Tengjiao 389, 399, 559  
 Wang, Weiping 24  
 Wang, Xiaolin 186  
 Wang, Xiaoling 439  
 Wang, Xiu-Kun 745  
 Wang, Yu 409  
 Wang, Ziyang 291  
 Wechsler, Katia 315  
 Wu, Bin 107  
 Wu, Cen 696  
 Wu, Qing 107  
 Wu, Quanyuan 664  
 Wu, Shengli 303  
 Wu, Xuejun 646  
 Wu, Zhaohui 107
- Xing, Chunxiao 157  
 Xiong, Fang 549  
 Xu, Jian 34

Xu, Xiaofei 589, 726  
 Xu, Yaoqiang 157  
 Xu, Zhuoqun 186  
 Xue, Gui-Rong 208  
  
 Yang, Dongqing 389, 399, 559  
 Yang, Xiaochun 478  
 Yao, Tianshun 646  
 Ye, Na 646  
 Ye, Yunming 489  
 Yeh, Hongjin 87  
 Yu, Ge 478, 600, 739  
 Yu, Hong 745  
 Yu, Jeffrey Xu 147, 549  
 Yu, Man Hing 429  
 Yu, Shou-Jian 45  
 Yu, Yong 208  
 Yuan, Chang-an 55  
 Yuan, Yang 439  
 Yue, Kun 439  
 Yuen, Man-Ching 127

Zhang, Dongdong 24  
 Zhang, Huaxiang 628  
 Zhang, Jian-Ying 745  
 Zhang, Liang 167  
 Zhang, Xiao-Di 640  
 Zhang, Yanqiu 634, 690, 702  
 Zhang, Zhaogong 24, 702  
 Zhang, Zheng 509  
 Zhao, Faxin 600  
 Zhao, Kai 634  
 Zhou, Aoying 368, 439, 509, 549  
 Zhou, Lizhu 157  
 Zhou, Shuigeng 509  
 Zhou, Xiaofang 34  
 Zhu, Liang 197  
 Zhu, Jingbo 646  
 Zhu, Xing 208  
 Zhuge, Hai 13  
 Zou, Futai 167  
 Zuo, Jie 55