Lecture Notes in Bioinformatics 4983

Edited by S. Istrail, P. Pevzner, and M. Waterman

Subseries of Lecture Notes in Computer Science

Ion Măndoiu Raj Sunderraman
Alexander Zelikovsky (Eds.)

Bioinformatics Research and Applications

Fourth International Symposium, ISBRA 2008
Atlanta, GA, USA, May 6-9, 2008
Proceedings

Springer

Volume Editors

Ion Măndoiu
University of Connecticut
Computer Science and Engineering Department
Storrs, CT 06269, USA
E-mail: ion@engr.uconn.edu

Raj Sunderraman
Alexander Zelikovsky
Georgia State University
Computer Science Department
Atlanta, GA 30303, USA
E-mail: {raj,alexz}@cs.gsu.edu

Library of Congress Control Number: 2008925339

CR Subject Classification (1998): J.3, H.2.8, F.1, F.2.2, G.3

LNCS Sublibrary: SL 8 – Bioinformatics

ISSN 0302-9743
ISBN-10 3-540-79449-2 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-79449-3 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 12262007 06/3180 5 4 3 2 1 0

Preface

The 4^{th} edition of the International Symposium on Bioinformatics Research and Applications (ISBRA 2008) was held on May 6-9, 2008 at Georgia State University in Atlanta, Georgia. The symposium provides a forum for the exchange of ideas and results among researchers, developers, and practitioners working on all aspects of bioinformatics and computational biology and their applications.

The technical program of the symposium included 35 contributed papers, selected by the Program Committee from a number of 94 full submissions received in response to the call for papers. The technical program also included six papers contributed to the First International Workshop on Optimal Data Mining in Gene Expression Analysis (ODGEA 2008), which was held in conjunction with ISBRA 2008. In addition to the contributed papers, the symposium included tutorials and poster sessions and featured invited keynote talks by six distinguished speakers. Andrew Scott Allen from Duke University and Dan Nicolae from the University of Chicago spoke on novel analysis methods for genome-wide association studies; Kenneth Buetow, director of the National Cancer Institute Center for Bioinformatics, spoke on the cancer Biomedical Informatics Grid; Andrey Gorin from Oak Ridge National Laboratory spoke on peptide identification from mass spectrometry data; Yury Khudyakov from the Center for Disease Control and Prevention spoke on integrative viral molecular epidemiology; and Kwok Tsui from Georgia Institute of Technology spoke on data mining and statistical methods for analyzing microarray experiments.

We would like to thank the Program Committee members and external reviewers for volunteering their time to review and discuss symposium papers. We would also like to thank the Chairs and the Program Committee of ODGEA 2008 for enriching the technical program of the symposium with a workshop on an important and active area of bioinformatics research. We would like to extend special thanks to the General Chairs of the symposium for their continued leadership, and to the Local Organization, Publications, Finance, Publicity, and Posters Chairs for their hard work in making ISBRA 2008 a successful event. Last but not least we would like to thank all authors for presenting their work at the symposium.

May 2008

Ion Măndoiu
Raj Sunderraman
Alexander Zelikovsky

Symposium Organization

General Chairs

Dan Gusfield, University of California, Davis, USA
Yi Pan, Georgia State University, USA

Program Chairs

Ion Mandoiu, University of Connecticut, USA
Raj Sunderraman, Georgia State University, USA
Alexander Zelikovsky, Georgia State University, USA

Local Organization Chairs

Robert Harrison, Georgia State University, USA
Yanqing Zhang, Georgia State University, USA

Workshop Chairs

Luonan Chen, Osaka Sangyo University, Japan
Lonnie Welch, Ohio University, USA

Finance Chairs

Anu Bourgeois, Georgia State University, USA
Akshaye Dhawan, Georgia State University, USA

Publicity Chairs

Dumitru Brinza, University of California, San Diego, USA
Yingshu Li, Georgia State University, USA

Poster Chairs

Gulsah Altun, Georgia State University, USA
Stefan Gremalschi, Georgia State University, USA

Website

Web Design: Diana Mohan Babu, University of California, San Diego, USA
Web Master: Gulsah Altun, Georgia State University, USA

Program Committee

Gabriela Alexe
Broad Institute of MIT and Harvard, USA

Yonatan Aumann
Bar Ilan University, Israel

Zvi Bar-Joseph
Carnegie Mellon University, USA

Danny Barash
Ben-Gurion University, Israel

Anne Bergeron
Université du Québec à Montréal, Canada

Tanya Berger-Wolf
University of Illinois at Chicago, USA

Piotr Berman
Pennsylvania State University, USA

Olivier Bodenreider
National Library of Medicine, NIH, USA

Jeremy Buhler
Washington University in St. Louis, USA

Paola Bonizzoni
Univ. de Studi di Milano-Bicocca, Italy

Mark Borodovsky
Georgia Institute of Technology, USA

Daniel Brown
University of Waterloo, Canada

Liming Cai
University of Georgia, USA

Doina Caragea
Kansas State University, USA

Luonan Chen
Osaka Sangyo University, Japan

Yixin Chen
Washington University, USA

Peter Damaschke
Chalmers University, Sweden

Bhaskar Dasgupta
University of Illinois at Chicago, USA

Sorin Draghici
Wayne State University, USA

Colin Dewey
University of Wisconsin-Madison, USA

Liliana Florea
George Washington University, USA

Andrey Gorin
Oak Ridge National Laboratory, USA

Jun-Tao Guo
University of North Carolina at Charlotte, USA

Jieyue He
Southeast University, China

Vasant Honavar
Iowa State University, USA

Hae-Jin Hu
National Cancer Center, South Korea

Xiaohua (Tony) Hu
Drexel University, USA

Ming-Yang Kao
Northwestern University, USA

Marek Karpinski
University of Bonn, Germany

George Karypis
University of Minnesota, USA

John Kececioglu
University of Arizona, USA

Ed Keedwell
University of Exeter, UK

Yury Khudyakov
CDC, USA

Sun Wing Kin
NUS, Singapore

Istvan Ladunga
University of Nebraska-Lincoln, USA

Guojun Li
University of Georgia, USA

Jing Li
Case Western Reserve University, USA

Yiming Li
National Chiao Tung University, Taiwan

Guohui Lin
University of Alberta, Canada

Shiyong Lu
Wayne State University, USA

Jingchu Luo
Peking University, China

Osamu Maruyama
Kyushu University, Japan

Satoru Miyano
University of Tokyo, Japan

Bernard Moret
Ecole Poly. Fed. de Lausanne, Switzerland

Kayvan Najarian
Virginia Commonwealth University, USA

Giri Narasimhan
Florida Int. University, USA

Craig Nelson
University of Connecticut, USA

Laxmi Parida
IBM T.J. Watson Res. Ctr, USA

Andrey A. Perelygin
Georgia State University, USA

Itsik Pe'er
Columbia University, USA

Mihai Pop
University of Maryland, USA

Alex Pothen
Old Dominion University, USA

Teresa Przytycka
NCBI, USA

Sven Rahmann
Technical University of Dortmund, Germany

Sanguthevar Rajasekaran
University of Connecticut, USA

Ben Raphael
Brown University, USA

David Sankoff
University of Ottawa, Canada

Russell Schwartz
Carnegie Mellon University, USA

Hagit Shatkay
Queen's University, Canada

Jens Stoye
Universität Bielefeld, Germany

External Reviewers

Cen Gao
Yonghua Han
Mohammad Hasan
Petteri Hintsanen
Yang Huang
Peter Husemann
Katharina Jahn
Soo-Yeon Ji
Crystal Kahn
Chris Kauffman
RynagGuk Kim
Weimin Li
Yong Li
Tien-ho Lin
Yong Lu
Chunmei Liu
Jingping Liu
Peng Liu
Tobias Marschall
Jonathan Myers
Shilu Ni
Xia Ning
Vinhthuy Phan
Yanjun Qi
Anna Ritz
Saeed Salem
Petteri Sevon
Quanhu Sheng
Itamar Simon
Rebecca Smith
Yinglei Song
Anuj Srivastava
Deborah Stoffer
Gianluca Della Vedova
Lindi Wahl
Nikil Wale
Xiyin Wang
Roland Wittler
Yong Wu
You Xu
Zhiqiang Ye
Weizhe Zhang

Workshop Organization

First International Workshop on Optimal Data Mining in Gene Expression Analysis (ODGEA 2008)

General Chair

Lonnie R. Welch, Ohio University, USA

Program Chairs

R. Krishna Murthy Karuturi, Genome Institute of Singapore, Singapore
Qihua Tan, Odense University Hospital, Denmark
Jahangheer Shareef Shaik, University of Memphis, USA

Program Committee

Jing Hua Zhao, Medical Research Council in Cambridge, UK
Xiaoli Li, Institute of Infocom Research, A-Star, Singapore
Chengpeng Bi, University of Missouri - Kansas City, USA
Talamuthu Anbupalam, Genome Institute of Singapore, A-Star, Singapore
Vladimir Kuznetsov, Bioinformatics Institute, A-Star, Singapore
Meena Sakharkar, Nanyang Technological University, Singapore
Zhengxin Chen, University of Nebraska, Omaha, USA
Vinhthuy Phan, University of Memphis, USA
Parvathi Chundi, University of Nebraska, Omaha, USA
Lih Deng, University of Memphis, USA
Sach Mukherjee, University of Warwick, UK
Yuri Orlov, Genome Institute of Singapore, A-Star, Singapore
Vinsensius Vega, Genome Institute of Singapore, A-Star, Singapore

Table of Contents

Invited Keynote Talk: Set-Level Analyses for Genome-Wide Association Data

Dan L. Nicolae[1], Omar De la Cruz[2], William Wen[2], Baoguan Ke[2], and Minsun Song[2]

[1] Departments of Medicine and Statistics, The University of Chicago
[2] Department of Statistics, The University of Chicago
5734 S. University Ave., Chicago, IL 60637

High-throughput genotyping platforms allow the investigation of hundreds of thousands of markers at a time, and this has led to a growing number of genome-wide association studies in which the entire human genome is mined for genes involved in etiology of complex traits. This approach for discovery of genetic risk factors has yielded promising results, but most of the analyses have focused on single marker tests. In general, a method of analysis that uses the markers as if they are biologically unrelated throws away all the information contained in the structure of the genome.

In this paper, we propose a method for incorporating structural genomic information by grouping the markers in relevant units, and assigning a measure of significance to these pre-defined sets of markers. The sets can be genes, conserved regions, or groups of genes such as pathways. Using the proposed methods and algorithms, evidence for association between a particular functional unit and a disease status can be obtained not just by the presence of a strong signal from a SNP within it, but also by the combination of several simultaneous weaker signals that are uncorrelated. Note that the method will combine evidence for association from both the genotyped and the untyped markers. The untyped markers are tested using haplotype predictors for their alleles, with the prediction training done in reference databases such as HapMap.

There are several advantages in using this approach. There is an increase in the power of detecting genes associated to disease because moderately strong signals within a gene are combined to obtain a much stronger signal for the gene as a functional unit. The results are easily combined across platforms that use different sets of SNP. Lastly, the results are easy to interpret since the refer to functional regions, and they also provide targets for biological validation.

I. Măndoiu, R. Sunderraman, and A. Zelikovsky (Eds.): ISBRA 2008, LNBI 4983, p. 1, 2008.

Hierarchical Clustering Using Constraints

Mariana Kant[1], Maurice LeBon[1], and David Sankoff[2]

[1] Computer Science and Engineering Department,
York University, Toronto, Canada M3J 1P3
mkant@yorku.ca
[2] Department of Mathematics and Statistics,
University of Ottawa, Ottawa, Canada K1N 6N5
sankoff@uottawa.ca

Abstract. We describe a new supertree algorithm that extends the type of information that can be used for phylogenetic inference. Its input is a set of constraints that expresses either the hierarchical relationships in a family of given phylogenies, or/and other relations between clusters of sets of species. The output of the algorithm is a multifurcating rooted supertree which satisfies all constraints. Moreover, if there were contradictions in the set of constraints the corresponding part of the supertree is identified and its set of constraints is displayed such as the user may decide to modify or keep it. Our algorithm is not affected by the order in which the input phylogenies or other constraints are presented. We apply our method to a number of data sets.

1 Introduction

Supertree construction has taken on increasing importance with the widespread adoption by biologists of the "Tree of Life" endeavour (e.g., [1,2,3,4]). The input to the supertree problem is a family $\mathcal{T}$ of rooted trees with overlapping leaf sets. The object is to build T, a supertree compatible with all trees in $\mathcal{T}$, namely a rooted tree whose leaf set includes all species of input trees and from which each tree of the family can be derived by a sequence of edge contractions.

Numerous algorithms have been proposed to build supertrees. Gordon [5] presented an algorithm for building a "strict consensus supertree" of two rooted binary trees. The time complexity of his algorithm is $O(p^3)$, where p is the maximum between the number of leaves of the two trees. Semple and Steel [6] proposed MinCutSupertree algorithm to build a rooted supertree for a family of rooted weighted binary trees. Berry and Nicholas [7] propose MergeTrees, an algorithm for constructing a supertree compatible with two binary trees by grafting "specific subtrees" or "specific leaves" of one tree onto the other. For $|\mathcal{T}| \geq 3$, they apply MergeTrees repeatedly to pairs of trees, each time reducing $|\mathcal{T}|$ by 1.

In this paper we reformulate the supertree problem in a general way, but so that it allows a unique solution by means of an efficient algorithm, which we detail. The input is a set of constraints, which may or may not be derived from trees, on sets of elements. There may be any number of constraints, e.g. for $|\mathcal{T}| \geq 3$. We derive the unique rooted supertree compatible (in a strong sense) with the set of

I. Măndoiu, R. Sunderraman, and A. Zelikovsky (Eds.): ISBRA 2008, LNBI 4983, pp. 2–13, 2008.

constraints. Rather than weaken the notion of compatibility, we allow multifurcating trees (cf. [8]), both for input and output, where contradictory constraints are handled by multifurcations. The output is independent of the order in which the constraints are presented in the input. The algorithm runs in time proportional to the number of distinct leaves across all constraints and proportional to the number of constraints. The algorithm identifies contradictions in the set of constraints, allowing the user to intervene and decide which constraints to retain or discard from the set, depending on the degree of resolution desired.

We illustrate our procedures with data on primate phylogeny.

2 Definitions

In this paper we use "tree" to denote a multifurcating rooted tree, as exemplified in Figure 1. Following the notation and terminology of [7], a tree T has a leaf set $L(T)$ in bijection with a label set. Each internal node (including the root) has at least two children. An edge between two internal nodes is an internal edge. A leaf $x, x \in L(T)$ is a descendant of an internal node u, if the path from x to the root passes through u. For a node u in T, we write $S(u)$ for the subtree rooted at u, i.e., u and all its descendant nodes, and $L(u)$ the leaves of this subtree.

Definition 1 (Restriction of a tree). *The restriction of a tree T to a set of leaves X, itself a tree denoted $T|X$, is the smallest induced graph of T connecting leaves with labels in $X \cap L(T)$, where each degree two (non-root) node x as well as its two incident edges (u, x) and (v, x) are replaced by a single edge (u, v) to make the tree homeomorphically irreducible. If $\mathcal{T}$ is a collection of trees, then the collection of subtrees $\mathcal{T}|X := \{T|X : T \in \mathcal{T}\}$.*

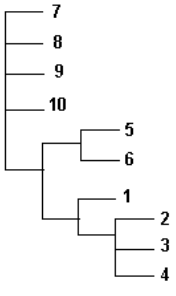


Fig. 1. Diagram of the tree T'

Definition 2 (Tree isomorphism and inclusion). *Two rooted trees T, T' are isomorphic, denoted $T = T'$, if and only if there is a graph isomorphism $T \to T'$ preserving leaf labels (and the root). Tree T is homeomorphically included in T' if and only if $T = T'|L(T)$.*

Definition 3 (Tree refinement). *If $L(T) = L(T')$, the tree T refines the tree T', written $T \ni T'$, if T can be transformed into T' by collapsing some of its internal edges (collapsing an edge means removing it and merging its extremities).*

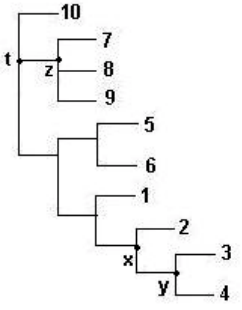


Fig. 2. Diagram of the tree T. T refines the tree T' in Figure 1.

The tree T in Figure 2 refines the tree T' in Figure 1. T' may be obtained from T by collapsing the edges (x, y) and (z, t).

Definition 4 (Tree compatibility). *Let T, T' be trees with leaf sets L, L', respectively. We say that T displays T' if $T|L' \supset T'$. Given a collection of trees $\mathcal{T}$, if there is a tree T that displays every tree in $\mathcal{T}$, we say that T displays $\mathcal{T}$ and that the collection $\mathcal{T}$ is compatible.*

Our algorithms will make use a definition of a set-theoretic definition of a tree [9,10,11].

Definition 5 (Tree-like family). *Let S be a set of n elements. $\mathcal{T}_S$, a tree-like family on the set S is a family of nonempty subsets of S such that:*

1. $S \in \mathcal{T}_S$,
2. $\{e\} \in \mathcal{T}_S$ *for all* $e \in S$,
3. $\forall A, B \in \mathcal{T}_S, A \cap B \in \{A, B, \emptyset\}$.

Example 1. *Given $S = \{1, 2, 3, 4, 5, 6\}$, the collection $\mathcal{T}_S = \{S, \{1\}, \{2\}, \{3\}, \{4\}, \{5\}\{6\}, \{1, 2, 3\}, \{4, 5, 6\}, \{4, 5\}\}$ is a tree-like family.*

It is clear that $\mathcal{T}_S$ is a partially ordered set under set inclusion. Moreover, for every set $\{e\}$ the collection $A_{\{e\}} = \{A : A \in \mathcal{T}_S \text{ and } \{e\} \subseteq A\}$ is a totally ordered set.

For a tree-like family $\mathcal{T}_S$, we consider the graph $G_S = (E, V)$, such that for each set $A \in \mathcal{T}_S$ there is a corresponding node $v_A \in V$; for every pair of nodes v_A, v_B, with $A \subseteq B$ and no set C in the family with $A \subseteq C \subset B$, there is an edge between v_A and v_B.

Example 2. *The graph in Figure 3 is the graph associated with the tree-like family in Example 1.*

Property 1. *The graph G_S associated to a tree-like family is a tree.*

Proof: Let v_S be the root of the tree. For every leaf $v_{\{e\}}$ there is an unique path between the root v_S and $v_{\{e\}}$, namely the path corresponding to all sets in the collection $A_{\{e\}}$. Suppose there is a cycle between two distinct nodes v_A and v_B. This means that $A \subseteq B$ and also $B \subseteq A$. Hence, $A = B$, implying that v_A and v_B are not distinct, a contradiction. □

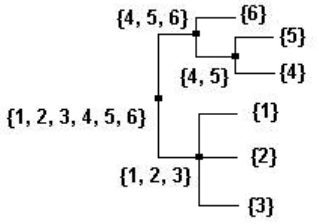


Fig. 3. Graph associated to a tree-like family. Also depicted are the sets associated to internal nodes.

Definition 6. *The tree-like family associated to the tree T, the set $\mathcal{T}_{L(T)}$ where* $\mathcal{T}_{L(T)} = \{L(T)\} \cup \{\{v\} : v \in L(T)\} \cup \{\{L(u)\} : u$ internal node of $T\}$.

Example 3. *The family* $\mathcal{T}_{L(T)} = \{\{1\}, \{2\}, \{3\}, \{4\}, \{5\}, \{6\}, \{7\}, \{8\}, \{9\}, \{10\}, \{1, \ldots, 10\}, \{3, 4\}, \{2, 3, 4\}, \{1, 2, 3, 4\}, \{5, 6\}, \{1, \ldots, 6\}, \{7, 8, 9\}\}$ *is the tree-like family associated to the tree T presented in Figure 2.*

Definition 7 (Constraints). *Let π denote a partition of a set S. We call every element of π, a block. The partition π satisfies the constraint $A \subseteq B$, where B is a subset of S, iff A is included in a block of π. Let C be a set of constraints. The partition π satisfies C if it satisfies every constraint in C. Given a tree T, and u, v, two internal nodes where u is a child of v, the constraint associated to the internal edge between u and v is $L(u) \subseteq L(v)$. We denote C_T the set of constraints for all internal edges of T.*

Example 4. *Given the set $S = \{1, 2, 3, 4, 5, 6\}$ and $\pi = \{\{1, 2, 3\}, \{4, 5, 6\}\}$ a partition on S. π satisfies the set of constraints $\{\{1, 2, 3\} \subseteq S, \{4, 5, 6\} \subseteq S, \{4, 5\} \subseteq \{4, 5, 6\}\}$.*

Example 5. *In Figure 2, the constraint $\{3, 4\} \subseteq \{2, 3, 4\}$ is associated with the edge connecting nodes x and y, and $\{7, 8, 9\} \subseteq \{1, 2, \ldots, 10\}$ to the edge connecting z and t. The set $C_T = \{\{3, 4\} \subseteq \{2, 3, 4\}, \{2, 3, 4\} \subseteq \{1, 2, 3, 4\}, \{5, 6\} \subseteq \{1, \ldots, 6\}, \{1, \ldots, 4\} \subseteq \{1, \ldots, 6\}, \{7, 8, 9\} \subseteq \{1, \ldots, 10\}, \{1, \ldots, 6\} \subseteq \{1, \ldots, 10\}\}$ is the set of constraints associated with T.*

Definition 8 (π_1 refines π_2). *For π_1, π_2 two partitions of the same set S, we say that π_1 refines π_2, denoted $\pi_1 \leq \pi_2$, iff for every pair A, B, with A a block of π_1 and B a block of π_2, the intersection set $A \cap B$ is in $\{A, \emptyset\}$.*

Example 6. *The partition $\pi_1 = \{\{1, 2\}, \{3, 4\}, \{5, 6\}, \{7, 8\}, \{9, 10\}\}$ refines the partition $\pi_2 = \{\{1, 2, 3, 4\}, \{5, 6\}, \{7, 8, 9, 10\}\}$.*

Definition 9 (π_2 embedded in π_1). *For two partitions π_1 and π_2 we say that π_2 is embedded in π_1 if π_2 is a partition of a block of π_1.*

Example 7. *The partition $\pi_1 = \{\{1, 2\}, \{3, 4\}\}$ is embedded in the partition $\pi_2 = \{\{1, 2, 3, 4\}, \{5, 6\}, \{7, 8, 9, 10\}\}$.*

Example 8. *Let S be the set with elements $\{1,2,3\}$ and C the set of constraints $\{\{1,2\} \subseteq \{1,2,3\}, \{1,3\} \subseteq \{1,2,3\}\}$. There is a contradiction between the two constraints: elements 1 and 2 cannot form a subcluster of $\{1,2,3\}$ at the same time as 1 and 3 form a subcluster of $\{1,2,3\}$. In this case the algorithm TREE-LIKE presented below can only return the partition containing single block $\{1,2,3\}$.*

The algorithm COMP presented in this paper is an extension of the procedure with the same name in [8]. Given a set of leaves S and a set of constraints C on S, the procedure COMP(S, C, π) iteratively builds π, a partition of S satisfying C. The same paper contains a procedure TREE$(S, C, \mathcal{T}_S)$ which constructs a family of embedded partitions $\pi_1, \pi_2 \ldots, \pi_k$ and returns $\mathcal{T}_S = \bigcup_{i=1}^{k} \pi_i$, where $\mathcal{T}_S$ is a tree-like family on S. Our algorithm TREE-LIKE below is an extension of procedure TREE.

3 Algorithms

We use Card(A) for the cardinality of set A.

Algorithm 1

Algorithm: COMP(*S, C,π)*
Input: A set of leaves *S*, and a set of constraints *C* on *S*.
Output: π, a partition of *S* that satisfies the set of constraints *C*.
begin
1. *count* ← 1
5. Let π^0=*{S₁, ..., S_k} , k=Card(S)*, be the initial partition of *S*, where S_i= *{s_i}, 1≤ i ≤k*, consists of a single element (leaf) of *S*.
3. $\pi \leftarrow \pi^0$
4. **repeat** while *count* ≠0
5. *count* ← 0
6. **for each** constraint $A \subseteq B$ in *C* **do**
7. Find all S_t in π such that $A \cap S_t \neq \emptyset$.
8. **if** there are at least two such S_t , **then**
($A \cap S_t \neq \emptyset$ for only one S_t means that $A \subseteq S_t$ and the constraint is already satisfied.)
9. *count* ← 1
10. $S_{new} \leftarrow \bigcup_{\substack{t=1,\\ S_t \cap A \neq \emptyset}}^{|\pi|} S_t$ // A ⊆ S_{new}
11. Delete from π all S_t with $A \cap S_t \neq \emptyset$.
// they will be replaced by their union.
12. Add S_{new} to π
13. **if** *Card(π)* =1 **then**
14. **if** *Card (S)* >1 **then** // there is a contradiction in *C*
15. **return** π
16. **end** (if)
17. **end**(if)
18. **end** (if)
19. **end** (for)
20. **end** (repeat)
21. **return** π
end

π returned by COMP(S, C, π) is a partition of S. The algorithm starts with π, a family of single element blocks, which is clearly a partition of S. During steps 6-12 one or more blocks of π are deleted and their union is added to the family as a new block S_{new}. Clearly $A \subseteq S_{\text{new}}$ and the number of blocks in the partition diminishes by at least one at each pass of steps 6-12. Consequently, π returned by algorithm COMP(S, C, π) is a partition which satisfies the constraint set C. If π has only one block and the set of constraints is not empty then the set of constraints is satisfied but it contains at least one contradiction.

Algorithm 2

Algorithm: TREE-LIKE(S, C, $\mathcal{T}_S$)
Input: A set of leaves S, and a set of constraints C on S.
Output: $\mathcal{T}_s$, a family of embedded partitions, such that each partition satisfies its corresponding constraints of C.
begin
1. $\mathcal{T}_s \leftarrow \emptyset$
Let $\pi^0=\{S_1, ..., S_k\}$, $k=Card(S)$, be the initial partition of S,
where $S_i=\{s_i\}$, $1 \le i \le k$, consists of a single element (leaf) of S.
2. **if** $C=\emptyset$, **then** $\mathcal{T}_s \leftarrow \mathcal{T}_s \cup \pi^0$ // partition having each block with cardinal 1
3. **return** $\mathcal{T}_s$
else
4. Print the elements of S
5. Print the elements of C.
6. COMP(S, C, π)
Suppose $\pi = \{S_1, ..., S_m\}$, $m \ge 1$
7. $\mathcal{T}_s \leftarrow \mathcal{T}_s \cup \pi$
8. **if** $|\pi|=1$ **then**
9. Print "Possible error: The partition has only one block"
10. **return** $\mathcal{T}_s$
else
11. **for** i=1 **to** m **do**
12. $C_{Si} \leftarrow \{A \subseteq B: A \subseteq B \in C$ and $B \subseteq S_i\}$
(C_{Si} is the subset of C concerning only the elements of S_i.
so that Card(C_{Si}) ≤ Card(C).)
13. TREE_LIKE(S_i, C_{Si}, $\mathcal{T}_i$)
14. $\mathcal{T}_s \leftarrow \mathcal{T}_s \cup \mathcal{T}_i$
15. **end** (for)
16. **end** (if)
17. **end** (if)
18. **return** $\mathcal{T}_s$
end.

Suppose TREE-LIKE$(S_p, C_{S_p}, \mathcal{T}_p)$ and TREE-LIKE$(S_k, C_{S_k}, \mathcal{T}_k)$ are two distinct calls of TREE-LIKE during the execution of TREE-LIKE$(S, C, \mathcal{T}_S)$. The following situations may occur:

1. $S_p \cap S_k = \emptyset$, meaning that the two calls are made directly or recursively for distinct values of i in the for loop of TREE-LIKE$(S, C, \mathcal{T}_S)$;
2. $S_p \cap S_k \neq \emptyset$. Without loss of generality we suppose TREE-LIKE$(S_p, C_{S_p}, \mathcal{T}_p)$ calls directly or recursively TREE-LIKE$(S_k, C_{S_k}, \mathcal{T}_k)$. Hence, the partition $\mathcal{T}_k$ is embedded in the partition $\mathcal{T}_p$.

If there are no contradictions in the set of constraints then at each recursive call of TREE-LIKE the input contains at least one constraint less than the caller

TREE-LIKE. Hence, the last call on the recursive stack of executions is one with the constraints set empty. Consequently, the last partition returned is the partition where each block has cardinality 1.

If there are contradictions (Example 8) in the set of constraints, the algorithm will return a partition with a single block S' and will not continue the recursion. A message saying "Possible error. The partition has only one block" is displayed. To complete the algorithm, we also consider a subsequent partition for S' with single elements blocks.

Property 2. *$\mathcal{T}_S \cup \{S\}$ is a tree-like family.*

Proof: S belongs to the family. Every single leaf set is in the family. Consider A, B two distinct sets in the family that are blocks in two partitions built during the execution of the algorithm. Then $A \cap B \in \{A, B, \emptyset\}$.

4 Implementation

The algorithms COMP(S, C, π) and TREE-LIKE$(S, C, \mathcal{T}_S)$ were implemented using Java. We use a memory bit to represent each element of S and consequently we realized all operations on sets using the Bitwise operators of the BitSet Class. The result is a speed up of the running time of the program.[1]

Let k be the cardinal of the set of constraints C. The running time of the algorithm COMP(S, C, π) is $O(k)$ The set of constraints C is usually scanned only once. If there are many scans to be done, then the number of repetitions is no greater than k.

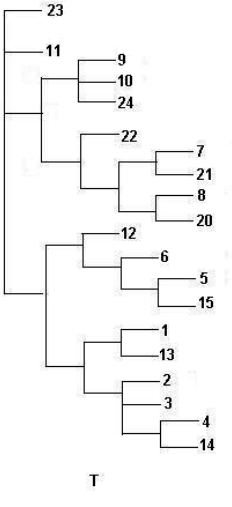


Fig. 4. T, supertree compatible with trees T_1, T_2 and T_3

[1] The program may be obtained from `mkant@yorku.ca`

The algorithm TREE-LIKE$(S, C, \mathcal{T}_S)$ is recursive. The total numbers of calls equals the number of internal nodes in the final supertree. Let p be the cardinality of the set of leaves in S. The worst-case running time of TREE-LIKE is then $O(kp)$. In the present implementation, at each recursive call of TREE-LIKE the program outputs the corresponding set of elements and constraints. Hence, the user may identify the contradictions in the set of constraints and choose which constraints to retain and which ones to discard.

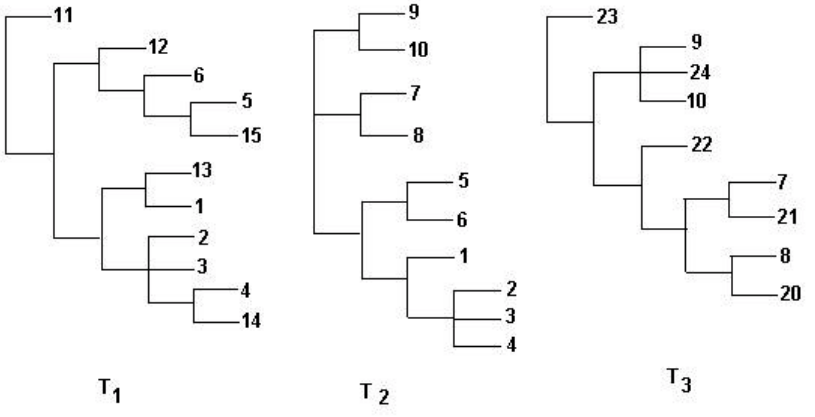


Fig. 5. Trees T_1, T_2 and T_3

5 Applications

5.1 Supertree for a Family of Trees with Refinement

Consider the trees T_1, T_2 and T_3 (Figure 5) where $L(T_1)$=$\{1, 2, 3, 4, 5, 6, 11, 12, 13, 14, 15\}$, $L(T_2) = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$, $L(T_3) = \{7, 8, 9, 10, 20, 21, 22, 23, 24\}$, where $T_1|L(T_1)\cap L(T_2)$ ϶ $T_2|L(T_1)\cap L(T_2)$ and $T_3|L(T_2)\cap L(T_3)$ ϶ $T_2|L(T_2)\cap L(T_3)$. The three trees (presented in Figure 5) have overlapping sets of leaves. Trees T_1 and T_3 include subtrees which are refinements of subtrees of T_2.

The tree T in Figure 4 is the supertree compatible with T_1, T_2 and T_3. It refines all of them: $T|L(T_2)$ ϶ $T2, T|L(T_1)$ ϶ T_1, and $T|L(T_3)$ ϶ T_3.

5.2 Supertree for a Family of Trees with Contradiction

Koop et al. [12] computed the independent phylogenies for the $\epsilon-, \gamma-, \eta-, \delta-$, and $\beta-$globin genes of groups of primates depicted in Figure 6. We obtained the supertree in Figure 7 for this family of phylogenies.

In the $\beta-$globin phylogeny {Chimpanzee, Gorilla} $\subseteq$ {Chimpanzee, Gorilla, Human} while in the $\eta-$globin phylogeny {Chimpanzee, Human}$\subseteq${Chimpanzee, Gorilla, Human}. This is a contradiction. Hence, the supertree computed by TREE-LIKE for the five phylogenies presents an internal node for the set {Chimpanzee, Gorilla, Human} with three children (Figure 7 (a), internal node x). To resolve this, we could choose, for example, to retain the constraint from

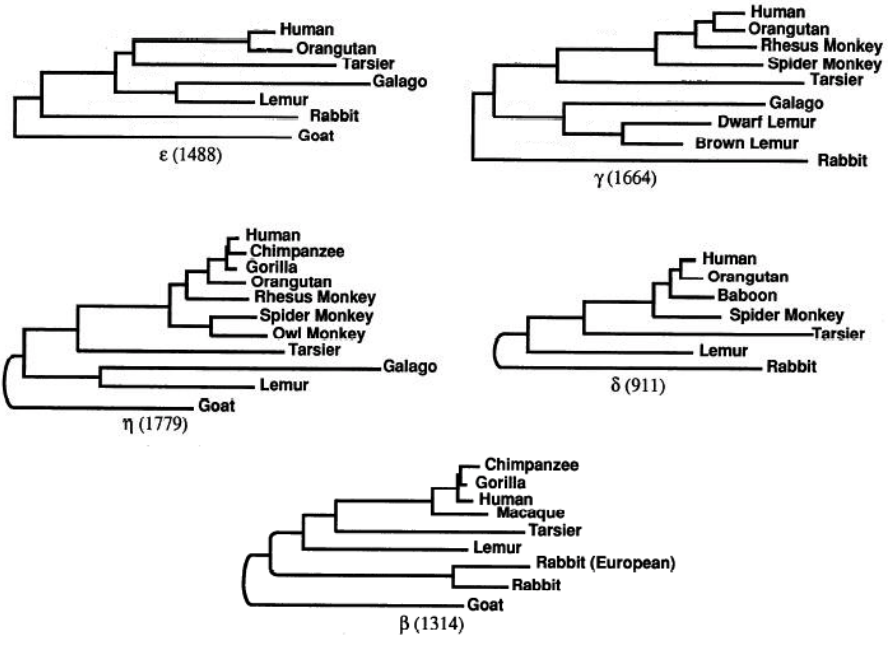


Fig. 6. Phylogenies computed for $\epsilon-, \gamma-, \eta-, \delta-$, and $\beta-$globin genes of groups of primates. From [12]. Used with permission.

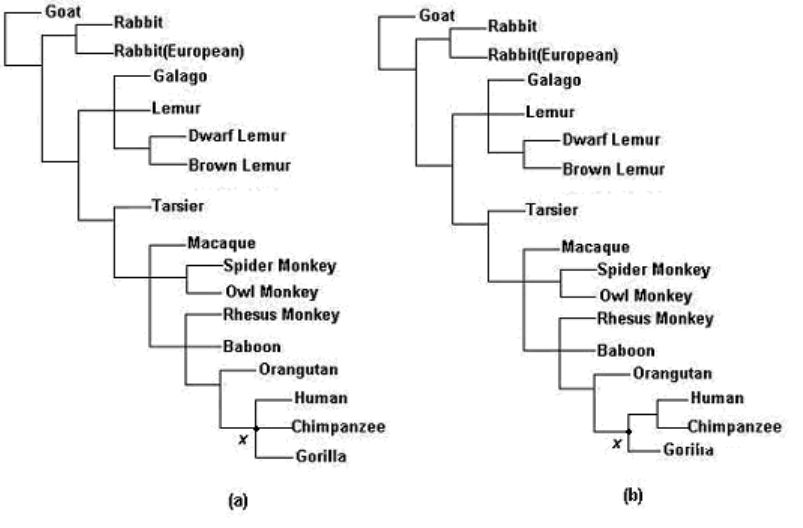


Fig. 7. Supertree for the phylogenies for $\epsilon-, \gamma-, \eta-, \delta-$, and $\beta-$globin genes of primates

the $\eta-$globin phylogeny and to discard the other. The output supertree would then have, from node x, a branch for Gorilla and another for the group Chimpanzee and Human (Figure 7 (b)).

5.3 Supertree for a Set of Constraints on Clusters

Consider the set of elements $L^* = \{1, \ldots, 14\}$ and the family of constraints $C^* = \{\{1,2\} \subseteq \{1,2,3\}, \{1,2,3\} \subseteq \{1,2,3,4,5,6\}, \{7,8\} \subseteq \{1,2,3,4,5,6,7,8\}, \{1,2,3,4,5,6,7,8\} \subseteq \{1,2,3,4,5,6,7,8,9,10\}, \{11,12\} \subseteq \{11,12,13,14\}, \{11,12,13,14\} \subseteq \{1,2,3,4,5,6,7,8,9,10,11,12,13,14\}\}$; $A \subseteq B$ meaning that either the elements in subset A are closer each other then the elements in B, or that the cluster A is a distinct subcluster of the cluster B of L^*.

The supertree T^* corresponding to the set of constraints C^* is presented in Figure 8(a). The set of constraints identifies $\{1,2,3,4,5,6,7,8\}$ as a subcluster of $\{1,2,3,4,5,6,7,8,9,10\}, \{1,2,3\}$ as subcluster of $\{1,\ldots 6\}, \{7,8\}$ as subcluster of $\{1,\ldots,8\}$. There is no specific information for elements 4, 5, and 6. Consequently, those three elements are presented as children of internal node x, Figure 8(a). Adding the constraint $\{4,5,6\} \subseteq \{1,2,3,4,5,6,7,8,9,10\}$ induces the algorithm to build the supertree in Figure 8(b).

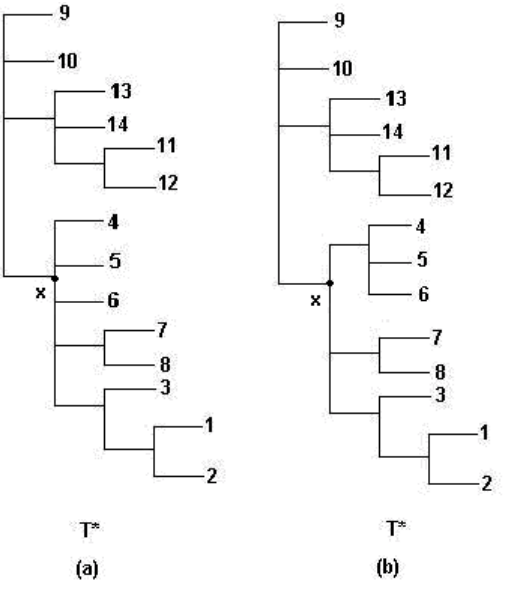


Fig. 8. Supertree obtained for a set of constraints on clusters

6 Conclusion

The input of our algorithm may be constraints derived from cluster analyses and/or from a family of small rooted trees with overlapping leaf sets or simply

from previous biological knowledge. It outputs a multifurcating supertree, which is the unique tree compatible in a strong sense with all the constraints, and which does not depend on any ordering of these constraints, either in the input or in preprocessing. At each recursive call of TREE-LIKE the program outputs the set of elements and the set of associated constraints. This information identifies contradictory constraints and helps the users decide if they must be retained, further resolving the tree, or not.

Semple and Steel [6] have written about desirable properties of supertrees. Among these are:

1. the method runs in polynomial time
2. the resulting supertree displays all rooted binary supertrees shared by all of the trees in the family
3. if the family is compatible, the resulting supertree displays each of the trees in the family
4. (a) the resulting supertree is independent of the order in which the members of the family are listed
   (b) if we rename all species and then apply the method to this new collection of input trees, the resulting supertree is the one obtained by applying the method to the original collection of trees, but with the species renamed as before.
5. the method allows a possible weighting of the trees in the family.

Of these properties, our method satisfies 1-4. If there is no contradiction among trees, property 5 is moot. If there is a contradiction, our provision for manual intervention is a way of assigning priorities among the input trees, though it is not a pre-assigned numerical weight.

Daniel and Semple [13] have used the notion of "semilabeling" in order to build supertrees that are based on incompatible trees. While this may well be advantageous from one point of view, it nevertheless has the disadvantage that the supertree can no longer display all the trees in the family on which it is based.

Acknowledgments

Research supported in part by grants from the Natural Sciences and Engineering Research Council of Canada (NSERC) to MK and to DS. DS holds the Canada Research Chair in Mathematical Genomics.

References

1. Cracraft, J., Donoghue, M.: Assembling the Tree of Life. Oxford University Press, Oxford (2004)
2. Bininda-Emonds, O.R.P.: Phylogenetic Supertrees: Combining Information to Reveal the Tree of Life, vol. 4. Kluwer Academic, Dordrecht (2004)

3. Hall, B.G.: Phylogenetic Trees Made Easy: A How-to Manual. 3 edn., Sinauer Associates, Inc (2007)
4. Bininda-Emonds, O.R.P., Gittleman, J.L., Steel, M.A.: The (Super) Tree of Life: Procedures, Problems, and Prospects. Annual Review of Ecology and Systematics 33, 265–289 (2002)
5. Gordon, A.D.: Consensus Supertrees: The Synthesis of Rooted Trees Containing Overlapping Sets of Labeled Leaves. Journal of Classification 3, 335–348 (1986)
6. Semple, C., Steel, M.: A Supertree Method for Rooted Trees. Discrete Applied Mathematics 105, 147–158 (2000)
7. Berry, V., Nicolas, F.: Maximum agreement and compatible supertrees. Journal of Discrete Algorithms 5, 564–591 (2007)
8. Constantinescu (Kant), M., Sankoff, D.: Tree Enumeration Modulo a Consensus. Journal of Classification 3, 349–356 (1986)
9. Margush, T., McMorris, F.R.: Consensus n-Trees. Bulletin of Mathematical Biology 43, 239–244 (1981)
10. Kant, M.: The Synthesis of Two Compatible Rooted Trees in a Rooted Supertree by an Algorithm on Sets. In: Proceedings of Fifth International Conference on Computing and Information (1993)
11. Day, W.H.E., Johnson, D.S., Sankoff, D.: The Computational Complexity of Inferring Rooted Phylogenies by Parsimony. Mathematical Biosciences 81, 33–42 (1986)
12. Koop, B.F., Tagle, D.A., Goodman, M., Slightom, J.L.: A molecular view of primate phylogeny and important systematic and evolutionary questions. Molecular Biology and Evolution 6, 580–612 (1989)
13. Daniel, P., Semple, C.: A class of general supertree methods for nested taxa. SIAM Journal of Discrete Mathematics 19, 463–480 (2005)

The Gene-Duplication Problem: Near-Linear Time Algorithms for NNI Based Local Searches

Mukul S. Bansal and Oliver Eulenstein

Department of Computer Science, Iowa State University, USA
{bansal, oeulenst}@cs.iastate.edu

Abstract. The gene-duplication problem is to infer a species supertree from a collection of gene trees that are confounded by complex histories of gene duplication events. This problem is NP-complete and thus requires efficient and effective heuristics. Existing heuristics perform a stepwise search of the tree space, where each step is guided by an exact solution to an instance of a local search problem. A classical local search problem is the NNI *search problem*, which is based on the nearest neighbor interchange operation. In this work we (i) provide a novel near-linear time algorithm for the NNI search problem, (ii) introduce extensions that significantly enlarge the search space of the NNI search problem, and (iii) present algorithms for these extended versions that are asymptotically just as efficient as our algorithm for the NNI search problem. The substantially extended NNI search problem, along with the exceptional speed-up achieved, make the gene-duplication problem more tractable for large-scale phylogenetic analyses.

1 Introduction

The rapidly increasing amount of available genomic sequence data provides an abundance of potential information for phylogenetic analyses. Most phylogenetic analyses combine genes from presumably orthologous loci, or loci whose homology is the result of speciation. These analyses largely neglect the vast amounts of sequence data from gene families, in which complex evolutionary processes such as gene duplication and loss, recombination, and horizontal transfer generate gene trees that differ from species trees. One approach to utilize the data from such gene trees (gene families) is to reconcile the gene trees with species trees based on the duplication optimality criterion that was introduced by Goodman et al. [13]. The corresponding optimization problem is called the *gene-duplication* problem [15]. This problem can be viewed as a *supertree problem*, that is, assembling from a collection of input trees (the gene trees) a species supertree that contains all species found in at least one of the input trees. The decision version of the gene-duplication problem is NP-complete [17]. Existing heuristics aimed at solving the gene-duplication problem search the space of all possible supertrees guided by a series of exact solutions to instances of a local search problem [20]. The local search problem is to find an optimal phylogenetic tree under the duplication optimality criterion in the neighborhood of a given tree. The neighborhood is the set of all phylogenetic trees into which the given tree can be transformed by applying a tree edit operation. A variety of different tree edit operations have been discussed in the literature [24, 26], and in practice the rooted nearest neighbor interchange (NNI) tree edit

I. Măndoiu, R. Sunderraman, and A. Zelikovsky (Eds.): ISBRA 2008, LNBI 4983, pp. 14–25, 2008.

operation has shown much potential for phylogenetic studies [15,22]. However, despite this potential, algorithms for local search problems based on NNI operations are still in their infancy. To conduct large-scale phylogenetic analyses, there is much need for more effective NNI based local search problems that can be solved efficiently.

In this work we extend the NNI neighborhood to the k–NNI neighborhood. The k–NNI neighborhood contains all trees that can be obtained by performing at most k successive NNI operations on the given tree.

Recently, efficient solutions were given for local search problems based on the standard SPR [2] and TBR [3] edit operations. It can be easily shown [11, 12] that 2 and 3–NNI neighborhoods of a tree have very small overlap with its SPR and TBR neighborhoods. This results in novel and potentially more effective local searches. We greatly improve on the complexity of the best known (brute-force) solutions for 2 and 3–NNI based local search problems. Furthermore, we show that each subsequent instance of the local search problem for 1, 2, and 3–NNI neighborhoods can be solved in linear time after the first instance is solved. This is especially desirable since standard local search heuristics for the gene-duplication problem typically involve solving several thousand instances of the local search problem. Our novel near-linear time algorithms provide much potential for making the gene-duplication problem more suitable for large-scale phylogenetic analyses.

1.1 Previous Results

The gene-duplication problem is based on the Gene Duplication model from Goodman et al. [13]. In the following, we (i) describe the Gene Duplication model, (ii) formulate the gene-duplication problem, and (iii) describe a heuristic approach of choice [20] to solve the gene-duplication problem.

Gene Duplication model. The Gene Duplication model is well studied [19, 15, 18, 29, 7, 5, 14] and explains incompatibilities between a pair of "comparable" gene and species trees through gene duplications. A gene and a species tree are *comparable*, if a *leaf-mapping* exists that provides a leaf to leaf mapping that maps every gene to the species from which it was sampled. The minimum number of gene duplications that are necessary under the Gene Duplication model to explain the incompatibilities can be inferred from the mapping M, which is an extension of the given leaf-mapping. M maps every gene in the gene tree to the most recent species in the species tree that could have contained the gene. More precisely, M maps each gene to the least common ancestor of the species from which the leaves (genes) of the subtree rooted at the gene were sampled (given by the leaf-mapping). A gene in the gene tree is a *duplication* if it has a child with the same M mapping. The *reconciliation cost* for a gene tree and a comparable species tree is measured in the number of gene duplications in the gene tree induced by the species tree. The *reconciliation cost* for a given collection of gene trees and a species tree is the sum of the reconciliation costs for each gene tree in the collection and the species tree. The mapping function is linear time computable on a PRAM [29] through a reduction from the least common ancestor problem [4]. Hence, the reconciliation cost for a collection of gene trees and a species tree is computable in linear time.

Gene-duplication problem and heuristics. The *gene-duplication problem* is to find for a given set of gene trees a comparable species tree with minimum reconciliation cost. This approach has been successfully applied to phylogenetic inference in snakes [27], vertebrates [21, 23], *Drosophila* [8], and plants [25] among others. However, the decision variant of this problem and some of its characterizations are NP-complete [17, 10] while some parameterizations are fixed parameter tractable [28, 16]. Therefore, in practice, heuristics (e.g. [20]) are commonly used for the gene-duplication problem, even though they are unable to guarantee an optimal solution. In these heuristics, a *tree graph* (see [1, 26]) is defined for the given set of gene trees and some fixed tree edit operation. Each node in the tree graph represents a unique species tree comparable with the given gene trees. An edge is drawn between two nodes exactly if the corresponding trees can be transformed into each other by one tree edit operation. The *reconciliation cost* of a node in the graph is the reconciliation cost of the species tree represented by that node and the given gene trees. Given an initial node in the tree graph, the heuristic's task is to find a maximal-length path of steepest descent in the reconciliation cost of its nodes and to return the last node on such a path. This path is found by solving the *local search problem* for every node along the path. The local search problem is to find a node with the minimum reconciliation cost in the neighborhood of a given node. The time complexity of the local search problem depends on the tree edit operation used.

Here, the edit operation of interest is the NNI operation [1, 6]. Rooted and unrooted NNI operations have been extensively studied [9]. An NNI operation on a species tree S (represented as an undirected graph) can be performed by "swapping" two of its node disjoint subtrees whose root nodes are connected by a simple path of length 3. The resulting tree graph is connected and every node has a degree of $\Theta(n)$, where n is the number of leaves in S. Thus, the local search problem for the k–NNI neighborhood and r gene trees can be solved naively in $O(rn^{k+1})$ time (assuming, for convenience, that the gene trees differ in size from the species tree by at most a constant factor). These brute-force solutions are the best available for $k \geq 1$, and hence, the development of faster algorithms is required in order to perform desired large scale phylogenetic studies using k–NNI local searches.

1.2 Contribution of This Work

We provide efficient algorithms for local search heuristics based on 1, 2 and 3–NNI neighborhoods. In fact, we show that local searches based on 2 and 3–NNI neighborhoods are asymptotically just as efficient as those based on 1–NNI, even though they search a much larger neighborhood of trees. For convenience assume that the size of the r given gene trees differs by a constant factor from the size of the resulting species tree, which we denote by n. Local searches based on 1, 2 and 3–NNI respectively induce neighborhoods of size $\Theta(n)$, $\Theta(n^2)$ and $\Theta(n^3)$; and hence, best known (brute-force) solutions for the 1, 2, and 3–NNI local search problems require $O(rn^2)$, $O(rn^3)$, and $O(rn^4)$ time respectively. We provide algorithms that solve the local search problems for both 2 and 3 NNI-neighborhoods in $O(rn^2)$ time.

Furthermore, we show that each subsequent 1, 2, or 3–NNI local search can be solved in $O(rn)$ time. In summary, for all three neighborhoods, the total complexity of a heuristic search involving p local search steps is $O(rn(n + p))$. Thus, if $p \geq n$,

which largely holds true in practice, then the amortized time complexity per local search step is linear in the input size. Consequently, our algorithms provide a total speed-up of $\Theta(\min\{n,p\})$, $\Theta(n \times \min\{n,p\})$, and $\Theta(n^2 \times \min\{n,p\})$ for heuristics that are based on 1, 2 and 3–NNI local searches respectively. It is interesting to note that for 2 and 3–NNI, the complexity of our algorithms is in fact sub-linear in the size of the corresponding neighborhoods. The substantially enlarged neighborhoods, and the exceptional speed-up achieved make the gene-duplication problem more tractable for large-scale phylogenetic analyses.

2 Basic Notation and Preliminaries

In this section we first introduce basic definitions and notation, and then the necessary preliminaries required for this work.

2.1 Basic Definitions and Notation

A *tree* T is a connected graph with no cycles, consisting of a node set $V(T)$ and an edge set $E(T)$. T is *rooted* if it has exactly one distinguished node called the *root* which we denote by $\mathrm{Ro}(T)$. Let T be a rooted tree. We define $\leq_T$ to be the partial order on $V(T)$ where $x \leq_T y$ if y is a node on the path between $\mathrm{Ro}(T)$ and x. The set of minima under $\leq_T$ is denoted by $\mathrm{Le}(T)$ and its elements are called *leaves*. If $\{x,y\} \in E(T)$ and $x \leq_T y$ then we call y the *parent* of x denoted by $\mathrm{Pa}_T(x)$ and we call x a *child* of y. The set of all children of y is denoted by $\mathrm{Ch}_T(y)$. If two nodes in T have the same parent, they are called *siblings*. The *least common ancestor* of a non-empty subset $L \subseteq V(T)$, denoted as $\mathrm{lca}(L)$, is the unique smallest upper bound of L under $\leq_T$. A *subtree* of T rooted at node $y \in V(T)$, denoted by T_y, is the tree induced by $\{x \in V(T) : x \leq y\}$. T is fully *binary* if every node has either zero or two children. Throughout this paper, the term tree refers to a rooted fully binary tree.

2.2 The Gene Duplication Problem

We now introduce necessary definitions to state the gene-duplication problem. A *species tree* is a tree that depicts the evolutionary relationships of a set of species. Given a gene family for a set of species, a *gene tree* is a tree that depicts the evolutionary relationships among the sequences encoding only that gene family in the given species. Thus, the nodes in a gene tree represent genes. In order to compare a gene tree G with a species tree S a mapping from each gene $g \in V(G)$ to the most recent species in S that could have contained g is required.

Definition 1 (Mapping). *A* leaf-mapping $\mathcal{L}_{G,S}\colon \mathrm{Le}(G) \to \mathrm{Le}(S)$ *specifies, for each gene g the species from which it was sampled. The extension $\mathcal{M}_{G,S}\colon V(G) \to V(S)$ of $\mathcal{L}_{G,S}$ is the* mapping *defined by $\mathcal{M}_{G,S}(g) = \mathrm{lca}(\mathcal{L}_{G,S}(\mathrm{Le}(G_g))$.*

Note: For any node $s \in V(S)$, $\mathcal{M}^{-1}_{G,S}(s)$ denotes the set of nodes in G that map to node $s \in V(S)$ under the mapping $\mathcal{M}_{G,S}$.

Definition 2 (Comparability). *The trees G and S are* comparable *if there exists a leaf-mapping $\mathcal{L}_{G,S}$. A set of gene trees $\mathcal{G}$ and S are* comparable *if each gene tree in $\mathcal{G}$ is comparable with S.*

Throughout this paper we use the following terminology: $\mathcal{G}$ is a set of gene trees that is comparable with a species tree S, and $G \in \mathcal{G}$.

Definition 3 (Duplication). *A node $v \in V(G)$ is a* (gene) duplication *if $\mathcal{M}_{G,S}(v) \in \mathcal{M}_{G,S}(\mathsf{Ch}(v))$ and we define $\mathsf{Dup}(G,S) = \{g \in V(G) \colon$ g is a duplication$\}$.*

Definition 4 (Reconciliation cost). *We define reconciliation costs for gene and species trees as follows:*

1. *$\Delta(G,S) = |\,\mathsf{Dup}(G,S)|$ is the* reconciliation cost from G to S.
2. *$\Delta(\mathcal{G},S) = \sum_{G\in\mathcal{G}} \Delta(G,S)$ is the* reconciliation cost *from $\mathcal{G}$ to S.*
3. *Let $\mathcal{T}$ be the set of species trees that is comparable with $\mathcal{G}$. We define $\Delta(\mathcal{G}) = \min_{S\in\mathcal{T}} \Delta(\mathcal{G},S)$ to be the* reconciliation cost of $\mathcal{G}$.

Problem 1 (Duplication)
Instance: *A set $\mathcal{G}$ of gene trees.*
Find: *A species tree S^* comparable with $\mathcal{G}$, such that $\Delta(\mathcal{G},S^*) = \Delta(\mathcal{G})$.*

2.3 Local Search Problems

Here we first provide the definition of an NNI edit operation [1, 6] and then formulate the related local search problems that were motivated in the Introduction.

Definition 5 (NNI operation). *Let T be a tree. For technical reasons we first define the set valid$(T) = V(T) \setminus \{\{\mathsf{Ro}(T)\} \cup \mathsf{Ch}(\mathsf{Ro}(T))\}$ and call its elements* valid nodes in T. *Now, for $y \in$ valid(T) we denote by $\mathsf{NNI}_T(y)$ the tree that is obtained from T by swapping the subtrees T_x and T_y where x is the sibling of $\mathsf{Pa}(y)$. We say that the tree $\mathsf{NNI}_T(y)$ is obtained from T by a* nearest neighbor interchange (NNI) operation on y *(an example is depicted in Fig. 1).*

In the remainder of this paper, whenever we write $\mathsf{NNI}_T(y)$ we assume that $y \in$ *valid*(T).

Definition 6 (k–NNI neighborhood). *The k–*NNI neighborhood *of a tree T is defined to be the set of all trees that can be obtained by performing at most k successive* NNI *operations on T. The k–*NNI *neighborhood of T is denoted by $k - \mathsf{NNI}_T$.*

Thus, for instance, $1 - \mathsf{NNI}_T$ (or simply NNI_T) is the set $\{\mathsf{NNI}_T(y) \colon y \in$ *valid*$(T)\}$.

Problem 2 (k–NNI–Search)
Instance: *A set $\mathcal{G}$ of gene trees, and a comparable species tree S.*
Find: *A tree $T^* \in k - \mathsf{NNI}_S$ such that $\Delta(\mathcal{G},T^*) = \min_{T\in k-\mathsf{NNI}_S} \Delta(\mathcal{G},T)$.*

In the next section we study the structural properties of 1, 2 and 3–NNI–Search problems. In Section 4 we develop our algorithm for 2–NNI–Search. Our algorithm for further speed-up of the p step 1 and 2–NNI heuristic search appears in Section 5. A description of our algorithm for the 3–NNI–Search problem, and its further speed-up appears in Section 6. Concluding remarks appear in Section 7.

3 Structural Properties

In the following we study the effects of an NNI operation on the mapping $\mathcal{M}_{G,S}$ and on the gene duplication status of nodes from G. Given G and S, consider an NNI operation that changes tree S into tree $S' = \mathrm{NNI}_S(y)$. Figure 1 depicts this situation. Figure 1 also depicts the naming convention that we follow for nodes in S before and after an NNI operation. Essentially, our naming convention preserves the name of each species tree node.

Note: In the interest of brevity, all lemmas in this paper appear with proofs omitted.

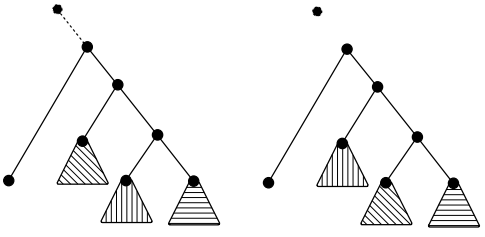


Fig. 1. The tree $S' = \mathrm{NNI}_S(y)$ is obtained by swapping the subtrees S_x and S_y

Lemma 1. *$\mathcal{M}^{-1}_{G,S}(s) = \mathcal{M}^{-1}_{G,S'}(s)$, for each $s \in V(S) \setminus \{u, v\}$ (see Figure 1).*

Definition 7. *For each $s \in valid(S)$ we define $diff_S(s) = \Delta(\mathcal{G}, S) - \Delta(\mathcal{G}, \mathrm{NNI}_S(s))$.*

Lemma 2. *Let $s \in valid(S)$, p and t be the siblings of s and $\mathrm{Pa}_S(s)$ in S, and p' and t' be the siblings of s and $\mathrm{Pa}_{S'}(s)$ in S' respectively. If $s \in valid(S')$, $\mathrm{Le}(S_t) = \mathrm{Le}(S'_{t'})$, $\mathrm{Le}(S_s) = \mathrm{Le}(S'_s)$, and $\mathrm{Le}(S_p) = \mathrm{Le}(S'_{p'})$, then $diff_S(s) = diff_{S'}(s)$.*

Definition 8. *Given $s \in valid(S)$, let a and b be the siblings of $\mathrm{Pa}_S(s)$ and s respectively. We define $ind_S(s) = valid(S) \setminus (\{a, b, s, \mathrm{Pa}_S(s)\} \cup \mathrm{Ch}_S(s) \cup \mathrm{Ch}_S(a) \cup \mathrm{Ch}_S(b))$, and say that the nodes in $ind_S(s)$ are independent with respect to node s in S.*

Essentially, the nodes in $ind_S(s)$ are important because they satisfy the property in Lemma 3. In the remainder of this paper, whenever we write $ind_S(s)$ we assume that $s \in valid(S)$. A key idea in our algorithms is that when an NNI operation is performed, much of the information computed for the original tree remains the same even for the new tree. This idea is formally captured in Lemma 3. It can be derived based on Lemma 2.

Lemma 3. *If $s \in valid(S') \cap ind_S(y)$, then $diff_{S'}(s) = diff_S(s)$.*

The next two lemmas follow more or less from the definition of $ind_S(s)$, and they are crucial for Lemma 6.

Lemma 4. *$|valid(S) \setminus ind_S(s)| \leq 10$.*

Lemma 5. *If $s \in valid(S)$, then $|\{t \in valid(S) \colon s \notin ind_S(t)\}| \leq 10$.*

4 Solving the 2–NNI–Search Problem

In this section we describe our algorithm to solve the 2–NNI–Search problem. The first step in our algorithm is to compute the value $\mathit{diff}_S(s)$ for each $s \in \mathit{valid}(S)$. This already gives a solution to the 1–NNI–Search problem. Subsequently, the algorithm computes a lowest reconciliation cost tree in $2-\mathrm{NNI}_S \setminus \mathrm{NNI}_S$. All trees in $2-\mathrm{NNI}_S \setminus \mathrm{NNI}_S$, are obtained by performing exactly 2 successive NNI operations on tree S. Consider some tree $T \in 2-\mathrm{NNI}_S \setminus \mathrm{NNI}_S$. Then there must exist two nodes $s,t \in V(S)$ such that $T = \mathrm{NNI}_{T'}(t)$, and $T' = \mathrm{NNI}_S(s)$. Now there are two possible cases:
(i) $t \in \mathit{ind}_S(s)$, or (ii) $t \notin \mathit{ind}_S(s)$.

The overall idea of our algorithm is as follows: We compute a minimum reconciliation cost tree among the trees that satisfy Case (i) above, and a minimum reconciliation cost tree among the trees that satisfy Case (ii). We also compute a minimum reconciliation cost tree in NNI_S. The best tree among these three trees must be a minimum reconciliation cost tree in $2-\mathrm{NNI}_S$. The lemmas that follow, allow us to efficiently compute a minimum reconciliation cost tree in $2-\mathrm{NNI}_S \setminus \mathrm{NNI}_S$.

Lemma 6. *Let A denote the set of the first 11 nodes valid in S arranged according to decreasing values of $\mathit{diff}_S(s)$. Let $\Gamma = \{T = \mathrm{NNI}_{T'}(t)\colon T' = \mathrm{NNI}_S(s)$, and $t \in \mathit{ind}_S(s)\}$. Let $R^* \in \Gamma$ with minimum reconciliation cost. Then, there exists a pair of nodes $a,b \in A$ such that $b \in \mathit{ind}_S(a)$, $R = \mathrm{NNI}_{R'}(b)$, $R' = \mathrm{NNI}_S(a)$, and $\Delta(\mathcal{G},R^*) = \Delta(\mathcal{G},R)$.*

Lemma 7. *Let $t \in \mathit{valid}(T)$ where $T = \mathrm{NNI}_S(s)$, such that $t \notin \mathit{ind}_S(s)$. Let a be the sibling of $\mathrm{Pa}_S(s)$, and b be the sibling of s in S. Then, $t \in \{\{a,b,s,\mathrm{Pa}_S(s)\} \cup \mathrm{Ch}_S(s) \cup \mathrm{Ch}_S(a) \cup \mathrm{Ch}_S(b)\}$.*

We can now present our algorithm to solve the 2–NNI–Search problem. We call this algorithm ALG-2-NNI , and a description of this algorithm appears as Algorithm 1.

The input for Algorithm 1 is a set of gene trees $\mathcal{G}$, and a species tree S. Let $n = |\,\mathrm{Le}(S)|$, and $r = |\mathcal{G}|$. To simplify the complexity analysis, we shall assume that all input gene trees have almost the same size. Thus, let $m = |\,\mathrm{Le}(S)| + |\,\mathrm{Le}(G)|$ for some $G \in \mathcal{G}$. Note: the speed-up obtained by our algorithm does not depend on this simplifying assumption.

Theorem 1. *Algorithm 1 solves the 2–NNI–Search problem in $O(rmn)$ time.*

Proof. (Correctness) Each tree $T \in 2-\mathrm{NNI}_S$ belongs to one of the following cases:

1. $T \in \mathrm{NNI}_S$: The tree T_1 computed in Algorithm 1 is a tree with minimum reconciliation cost among all trees in NNI_S.
2. $T \in 2-\mathrm{NNI}_S \setminus \mathrm{NNI}_S$: There exist two nodes $s,t \in V(S)$ such that $T = \mathrm{NNI}_{T'}(t)$, and $T' = \mathrm{NNI}_S(s)$. We now have two possible cases:
   (a) $t \in \mathit{ind}_S(s)$: According to Lemma 6, the tree T_2 computed by Algorithm 1 must be a minimum reconciliation cost tree among all trees in this case.
   (b) $t \notin \mathit{ind}_S(s)$: According to Lemma 7, the tree T_3 computed by Algorithm 1 must be a minimum reconciliation cost tree among all trees in this case.

Therefore, a minimum reconciliation cost tree among T_1, T_2, T_3 must be a solution to the 2–NNI–Search problem.

Algorithm 1. ALG-2-NNI

Input: A set of gene trees $\mathcal{G}$, and, a species tree S
Output: A tree $T^* \in k - \text{NNI}_S$ such that $\Delta(\mathcal{G}, T^*) = \min_{T \in k-\text{NNI}_S} \Delta(\mathcal{G}, T)$

1: **for** each $s \in \mathit{valid}(S)$ **do**
2: Compute the value $\mathit{diff}_S(s)$.
3: Let $\alpha \in \arg\max_{a \in \mathit{valid}(S)} \mathit{diff}_S(a)$, and set $T_1 = \text{NNI}_S(\alpha)$.
4: Let A denote the set of the first 11 nodes valid in S arranged according to decreasing values of $\mathit{diff}_S(s)$.
5: $(\alpha, \beta) \in \arg\max_{(a,b):\ a,b \in A,\ b \in \mathit{ind}_S(a)} \mathit{diff}_S(a) + \mathit{diff}_S(b)$.
6: Set $T = \text{NNI}_S(\alpha)$ and $T_2 = \text{NNI}_T(\beta)$.
7: Set $T_3 = T_2$.
8: **for** each $s \in \mathit{valid}(S)$ **do**
9: Let a be the sibling of $\text{Pa}_S(s)$, and b be the sibling of s in S. Set $T = \text{NNI}_S(s)$.
10: **for** $t \in \mathit{valid}(T) \cap \{\{a, b, s, \text{Pa}_S(s)\} \cup \text{Ch}_S(s) \cup \text{Ch}_S(a) \cup \text{Ch}_S(b)\}$ **do**
11: $R = \text{NNI}_T(t)$.
12: **if** $\Delta(\mathcal{G}, T_3) > \Delta(\mathcal{G}, T)$ **then**
13: Set $T_3 = T$.
14: **return** an element of $\arg\min_{T \in \{T_1, T_2, T_3\}} \Delta(\mathcal{G}, T)$.

(Complexity) Computing the tree T_1 involves computing the $\mathit{diff}_S(s)$ value for each $s \in \mathit{valid}(S)$, and identifying the node a for which $\mathit{diff}_S(a)$ is maximum. Computing the reconciliation cost for a given species tree takes $O(rm)$ time. Therefore, computing T_1 takes $O(rmn)$ time.

After T_1 has been computed, computing the tree T_2 involves creating the set A (which takes $O(n)$ time), and then evaluating every possible 2-element ordered pair from A. Each evaluation takes $O(1)$ time, and the number of possible ordered pairs is $O(|A|^2)$ i.e. $O(1)$. Therefore, computing T_2 (after having computed T_1) requires $O(n)$ time.

Computing T_3 involves evaluating the reconciliation costs of at most $10 \times n$ i.e. $O(n)$ trees, and then picking the best tree among these. Therefore, computing T_3 requires $O(rmn)$ time.

In conclusion, the time complexity of Algorithm 1 is $O(rmn)$. □

5 Further Speed-Up for 1 and 2–NNI Heuristics

As mentioned earlier, standard local search heuristics for the Duplication problem, involve solving many instances of these local search problems. Consider a heuristic search involving p instances of the local search problem, then, using our faster algorithm for the 2–NNI–Search problem allows both 1 and 2–NNI based heuristics to run in $\Theta(prmn)$ time. We will now show that the 1, 2–NNI based heuristics can, in fact, both be executed in $O(rm(n + p))$ time.

5.1 Heuristics Based on 1–NNI

Existing algorithms for the 1–NNI–Search (or simply NNI–Search) problem have a time complexity of $O(rmn)$, and hence they solve the NNI based heuristic problem

in $O(rpmn)$ time. Our algorithm to solve the NNI–Search problem involves computing the the value $\mathit{diff}_S(s)$ for each $s \in \mathit{valid}(S)$, and then picking a tree T such that $T = \text{NNI}_S(\alpha)$ where $\alpha = \arg\max_{a \in \mathit{valid}(S)} \mathit{diff}_S(a)$. This also requires $O(rmn)$ time. However, this approach allows us to reuse most of the previously computed information in subsequent iterations of the local search.

Let T denote a minimum reconciliation cost tree in NNI_S. Then, there exists a node a such that $T = \text{NNI}_S(a)$. For the next iteration we must compute a minimum reconciliation cost tree in NNI_T. As seen earlier, this involves computing the value $\mathit{diff}_T(s)$ for each $s \in \mathit{valid}(T)$. Let $\Gamma = \mathit{valid}(T) \cap \mathit{ind}_S(a)$. Then, by Lemma 3 we know that $\mathit{diff}_T(s) = \mathit{diff}_S(s)$, for all $s \in \Gamma$. Therefore, for all $s \in \Gamma$ we can reuse the values from the previous iteration. In other words we must only compute the value $\mathit{diff}_T(s)$ for all $s \in \mathit{valid}(T) \setminus \Gamma$. It follows directly from Lemma 7 that if $\Phi = \mathit{valid}(T) \setminus \Gamma$, then $|\Phi| \leq 10$.

This means that for each subsequent iteration of the NNI local search, we must compute the reconciliation costs for at most 10 trees. Thus, once the first NNI local search problem has been solved in $O(rmn)$ time, each subsequent local search instance can be solved in $O(rm)$ time. This gives a total time complexity of $O(rm(n+p))$, which gives a speed-up by a factor of $\Theta(\min\{n,p\})$ over existing solutions.

5.2 Heuristics Based on 2–NNI

Let T denote a minimum reconciliation cost tree in $2-\text{NNI}_S$. For the next iteration of this local search, we wish to find a tree U with minimum reconciliation cost in $2-\text{NNI}_T$. According to our algorithm (see Algorithm 1) computing U involves computing the trees $T_1, T_2, T_3 \in 2-\text{NNI}_T$. We now show how to compute each of these three special trees in $O(rm)$ time by reusing previously computed information.

There exist two nodes a, b such that $T' = \text{NNI}_S(a)$ and $T = \text{NNI}_{T'}(b)$. Computing the tree T_1 involves computing the value $\mathit{diff}_T(s)$ for all nodes $s \in \mathit{valid}(T)$. Since a and b are known (from the previous iteration of the local search), the method used for 1–NNI above can be used to obtain the values $\mathit{diff}_{T'}(s)$ for all $s \in \mathit{valid}(T')$ in $O(rm)$ time. Once this is done, the same algorithm is reapplied to compute the values $\mathit{diff}_T(s)$ for all $s \in \mathit{valid}(T)$. This step also takes $O(rm)$ time. Hence, the tree T_1 can be computed in $O(rm) + O(rm)$ i.e. $O(rm)$ time.

Once all the $\mathit{diff}_T(s)$ values have been obtained for all $s \in \mathit{valid}(T)$, computing the tree T_2 takes $O(n)$ time (see the complexity analysis in the proof of Theorem 1).

In order to compute the tree T_3, we first compute the tree $\text{NNI}_T(s)$ for each $s \in \mathit{valid}(T)$ and then compute the scores for at most 10 trees derived from $\text{NNI}_T(s)$, for each $s \in \mathit{valid}(T)$ (see Algorithm 1). We will show how to efficiently obtain all these $O(10n)$ scores by reusing the scores computed in the previous iteration of the local search. It is sufficient to show how to obtain these scores for the tree $T' = \text{NNI}_S(a)$, because the exact same procedure can be applied again on T' to obtain the scores for the tree $T = \text{NNI}_{T'}(b)$.

Let c, d be two nodes such that $R' = \text{NNI}_{T'}(c)$ and $R = \text{NNI}_{R'}(d)$. Since we wish to compute the tree corresponding to T_3, we may assume that $d \notin \mathit{ind}_{R'}(c)$. There are three possible cases:

1. $c, d \in \mathit{ind}_S(a)$: Let $Q = \mathrm{NNI}_S(c)$. In this case we have the values $\mathit{diff}_S(c)$ and $\mathit{diff}_Q(d)$ computed from the previous iteration. Since $c \in \mathit{ind}_S(a)$, by Lemma 3 we have $\mathit{diff}_{T'}(c) = \mathit{diff}_S(c)$. It can be shown that there are $O(1)$ candidates for d such that $\mathit{diff}_{R'}(d) \neq \mathit{diff}_Q(d)$. Thus, in case $\mathit{diff}_{R'}(d) \neq \mathit{diff}_Q(d)$, by Lemma 5 there are only $O(1)$ candidates for c as well, and hence the score for each such pair can be computed in $O(rm)$ time. Otherwise, the previously computed scores can be reused, which takes $O(n)$ time. This gives a total time complexity of $O(rm)$.
2. $c \in \mathit{ind}_S(a)$, $d \notin \mathit{ind}_S(a)$: d may be either valid or invalid in S. If $d \notin \mathit{valid}(S)$, then there are no more than two candidates for d (since $d \in \mathit{valid}(R')$, and R' is obtained from S by no more than two NNI operations). Otherwise, there are $O(1)$ candidates each for d (see Lemma 4). Since $d \notin \mathit{ind}_{R'}(c)$, Lemma 5 implies that there are $O(1)$ candidates for c. Hence, we only need to compute $O(1)$ scores.
3. $c \notin \mathit{ind}_S(a)$: c may be either valid or invalid in S. If $c \notin \mathit{valid}(S)$, then there is exactly one candidate for c (since $c \in \mathit{valid}(T')$). Otherwise, there are at most 10 candidates each for c and d (see Lemma 4). Hence, we only need to compute $O(1)$ scores.

Thus, T_3 can be computed in $O(rm)$ time as well, which in turn implies that a minimum reconciliation cost tree in $2 - \mathrm{NNI}_T$ can be computed in $O(rm)$ time. This gives a total time complexity of $O(rm(n+p))$ for 2–NNI based heuristics, which gives a speed-up by a factor of $\Theta(n \times \min\{n, p\})$ over the naive solution.

6 Optimizing the 3–NNI–Search Problem

The main idea behind our algorithms for the 1 and 2–NNI–Search problems, as well as their speed-up, is that when an NNI operation is performed on a tree, it only affects the mapping in a small, constant sized region of the tree. Since the reconciliation cost depends only on the mapping from the gene trees, in the new species tree thus obtained, much of the information computed for the original tree remains valid. This idea applies equally well for solving the k–NNI–Search problem, for $k > 2$, but the algorithm becomes progressively more convoluted as k increases. However, for the special case of $k = 3$, the algorithm for 2–NNI–Search extends in a rather straightforward manner.

The trees in $3 - \mathrm{NNI}_S$ must be in at least one of $2 - \mathrm{NNI}_S$, or $3 - \mathrm{NNI}_S \setminus 2 - \mathrm{NNI}_S$. We have already seen how to obtain a minimum reconciliation cost tree in $2 - \mathrm{NNI}_S$. Therefore, the problem is to find a minimum reconciliation cost tree in $3 - \mathrm{NNI}_S \setminus 2 - \mathrm{NNI}_S$. All the trees in $3 - \mathrm{NNI}_S \setminus 2 - \mathrm{NNI}_S$, are obtained by performing exactly 3 successive NNI operations on tree S. Consider some tree $T \in 3 - \mathrm{NNI}_S \setminus 2 - \mathrm{NNI}_S$. Then there must exist three nodes $s, t, u \in V(S)$ such that $T = \mathrm{NNI}_{T'}(u)$, $T' = \mathrm{NNI}_{T''}(t)$, and $T'' = \mathrm{NNI}_S(s)$. We now have six cases, exactly one of which must be true.

1. $t \in \mathit{ind}_S(s), u \in \mathit{ind}_{T''}(t) \cap \mathit{ind}_S(s)$
2. $t \in \mathit{ind}_S(s), u \in \mathit{ind}_S(s) \setminus \mathit{ind}_{T''}(t)$
3. $t \in \mathit{ind}_S(s), u \in \mathit{ind}_{T''}(t) \setminus \mathit{ind}_S(s)$
4. $t \in \mathit{ind}_S(s), u \notin \mathit{ind}_{T''}(t) \cup \mathit{ind}_S(s)$
5. $t \notin \mathit{ind}_S(s), u \in \mathit{ind}_{T''}(t) \cap \mathit{ind}_S(s)$
6. $t \notin \mathit{ind}_S(s), u \notin \mathit{ind}_{T''}(t) \cap \mathit{ind}_S(s)$

If we can calculate a minimum reconciliation cost tree separately for each of these six cases, then the tree with minimum cost among these six trees will be a minimum reconciliation cost tree in $3 - \text{NNI}_S \setminus 2 - \text{NNI}_S$.

It can be shown that a minimum reconciliation cost tree can be obtained for each of the six cases in $O(rmn)$ time (details omitted for brevity). This gives us an $O(rmn)$ time algorithm for the 3–NNI–Search problem.

The algorithm used to obtain further speed-up for 2–NNI based heuristics also extends in a similar fashion to 3–NNI based heuristics. This gives a total time complexity of $O(rm(n+p))$ for the 3–NNI based heuristic.

7 Outlook and Conclusion

We introduced algorithms that significantly speed up NNI based local search heuristics for the duplication problem. These algorithms extend naturally to local search problems based on the *Edge Contraction and Refinement (ECR)* edit operation [11, 12]. Thus, heuristic searches involving p instances of the 1, 2, or 3–ECR–Search problems can all be completed in $O(rm(n+p))$ time as well.

Our algorithms form the basis for extremely efficient local search heuristics. In particular, our 2 and 3–NNI local search algorithms can greatly improve on the performance of classical 1–NNI local search heuristics, without sacrificing efficiency. The real power of our algorithms can be best exploited as part of a heuristic that mixes 1, 2, and 3–NNI local searches with SPR and TBR local searches (see [22]). Such a heuristic would be both fast and effective, which would enable much larger analyses to be performed within a reasonable time. In future work, these techniques might set base for algorithmic theory that identifies a much broader class of local search problems which can be solved more efficiently.

Acknowledgements. This work was supported in part by NSF grant no. 0334832. The authors also wish to thank the anonymous reviewers for their invaluable comments.

References

1. Allen, B.L., Steel, M.: Subtree transfer operations and their induced metrics on evolutionary trees. Annals of Combinatorics 5, 1–13 (2001)
2. Bansal, M.S., Burleigh, J.G., Eulenstein, O., Wehe, A.: Heuristics for the gene-duplication problem: A $\Theta(n)$ speed-up for the local search. In: Speed, T., Huang, H. (eds.) RECOMB 2007. LNCS (LNBI), vol. 4453, pp. 238–252. Springer, Heidelberg (2007)
3. Bansal, M.S., Eulenstein, O.: An $\Omega(n^2/\log n)$ speed-up of TBR heuristics for the gene-duplication problem. In: Giancarlo, R., Hannenhalli, S. (eds.) WABI 2007. LNCS (LNBI), vol. 4645, pp. 124–135. Springer, Heidelberg (2007)
4. Bender, M.A., Farach-Colton, M.: The LCA problem revisited. In: Gonnet, G.H., Viola, A. (eds.) LATIN 2000. LNCS, vol. 1776, pp. 88–94. Springer, Heidelberg (2000)
5. Bonizzoni, P., Vedova, G.D., Dondi, R.: Reconciling a gene tree to a species tree under the duplication cost model. Theor. Comput. Sci. 347(1-2), 36–53 (2005)
6. Bordewich, M., Semple, C.: On the computational complexity of the rooted subtree prune and regraft distance. Annals of Combinatorics 8, 409–423 (2004)

7. Chen, K., Durand, D., Farach-Colton, M.: Notung: a program for dating gene duplications and optimizing gene family trees. Journal of Computational Biology 7, 429–447 (2000)
8. Cotton, J.A., Page, R.D.M.: Tangled tales from multiple markers: reconciling conflict between phylogenies to build molecular supertrees. In: Phylogenetic Supertrees: Combining Information to Reveal the Tree of Life, pp. 107–125. Springer, Heidelberg (2004)
9. DasGupta, B., He, X., Jiang, T., Li, M., Tromp, J., Zhang, L.: On distances between phylogenetic trees. In: SODA, pp. 427–436 (1997)
10. Fellows, M., Hallett, M., Korostensky, C., Stege, U.: Analogs and duals of the mast problem for sequences and trees. In: Bilardi, G., Pietracaprina, A., Italiano, G.F., Pucci, G. (eds.) ESA 1998. LNCS, vol. 1461, pp. 103–114. Springer, Heidelberg (1998)
11. Ganapathy, G., Ramachandran, V., Warnow, T.: Better hill-climbing searches for parsimony. In: Benson, G., Page, R.D.M. (eds.) WABI 2003. LNCS (LNBI), vol. 2812, pp. 245–258. Springer, Heidelberg (2003)
12. Ganapathy, G., Ramachandran, V., Warnow, T.: On contract-and-refine transformations between phylogenetic trees. In: SODA, pp. 900–909 (2004)
13. Goodman, M., Czelusniak, J., Moore, G.W., Romero-Herrera, A.E., Matsuda, G.: Fitting the gene lineage into its species lineage. a parsimony strategy illustrated by cladograms constructed from globin sequences. Systematic Zoology 28, 132–163 (1979)
14. Górecki, P., Tiuryn, J.: On the structure of reconciliations. In: Lagergren, J. (ed.) RECOMB-WS 2004. LNCS (LNBI), vol. 3388, pp. 42–54. Springer, Heidelberg (2005)
15. Guigó, R., Muchnik, I., Smith, T.F.: Reconstruction of ancient molecular phylogeny. Molecular Phylogenetics and Evolution 6(2), 189–213 (1996)
16. Hallett, M.T., Lagergren, J.: New algorithms for the duplication-loss model. In: RECOMB, pp. 138–146 (2000)
17. Ma, B., Li, M., Zhang, L.: From gene trees to species trees. SIAM J. Comput. 30(3), 729–752 (2000)
18. Mirkin, B., Muchnik, I., Smith, T.F.: A biology consistent model for comparing molecular phylogenies. Journal of Computational Biology 2(4), 493–507 (1995)
19. Page, R.D.M.: Maps between trees and cladistic analysis of historical associations among genes, organisms, and areas. Systematic Biology 43(1), 58–77 (1994)
20. Page, R.D.M.: GeneTree: comparing gene and species phylogenies using reconciled trees. Bioinformatics 14(9), 819–820 (1998)
21. Page, R.D.M.: Extracting species trees from complex gene trees: reconciled trees and vertebrate phylogeny. Molecular Phylogenetics and Evolution 14, 89–106 (2000)
22. Page, R.D.M., Charleston, M.A.: From gene to organismal phylogeny: reconciled trees and the gene tree/species tree problem. Molec. Phyl. and Evol. 7, 231–240 (1997)
23. Page, R.D.M., Cotton, J.: Vertebrate phylogenomics: reconciled trees and gene duplications. In: Pacific Symposium on Biocomputing, pp. 536–547 (2002)
24. Page, R.D.M., Holmes, E.C.: Molecular evolution: a phylogenetic approach. Blackwell Science, Malden (1998)
25. Sanderson, M.J., McMahon, M.M.: Inferring angiosperm phylogeny from EST data with widespread gene duplication. BMC Evolutionary Biology 7 (suppl. 1), 3 (2007)
26. Semple, C., Steel, M.: Phylogenetics. Oxford University Press, Oxford (2003)
27. Slowinski, J.B., Knight, A., Rooney, A.P.: Inferring species trees from gene trees: A phylogenetic analysis of the elapidae (serpentes) based on the amino acid sequences of venom proteins. Molecular Phylogenetics and Evolution 8, 349–362 (1997)
28. Stege, U.: Gene trees and species trees: The gene-duplication problem in fixed-parameter tractable. In: WADS, pp. 288–293 (1999)
29. Zhang, L.: On a Mirkin-Muchnik-Smith conjecture for comparing molecular phylogenies. Journal of Computational Biology 4(2), 177–187 (1997)

A Distance-Based Method for Detecting Horizontal Gene Transfer in Whole Genomes

Xintao Wei[1], Lenore Cowen[1], Carla Brodley[1],
Arthur Brady[1], D. Sculley[1], and Donna K. Slonim[1,2]

[1] Department of Computer Science, Tufts University, 161 College Ave., Medford, MA 02155
[2] Department of Pathology, Tufts University School of Medicine, 245 Harrison Ave., Boston, MA 02111
{xwei0a,cowen,brodley,abrady,dsculley,slonim}@cs.tufts.edu

Abstract. As the number of sequenced genomes has grown, we have become increasingly aware of the impact of horizontal gene transfer on our understanding of genome evolution. Methods for detecting horizontal gene transfer from sequence abound. Among the most accurate are methods based on phylogenetic tree inference, but even these can perform poorly in some cases, such as when multiple trees fit the data equally well. In addition, they tend to be computationally intensive, making them poorly suited to genomic-scale applications. We introduce a new method for detecting horizontal transfer that incorporates the *distances* typically used by phylogeny-based methods, rather than the trees themselves. We demonstrate that the distance method is scalable and that it performs well precisely in cases where phylogenetic approaches struggle. We conclude that a distance-based approach may be a valuable addition to the set of tools currently available for identifying horizontal gene transfer.

1 Introduction

Horizontal or lateral gene transfer, the transfer of genes between genomes rather than by "vertical" inheritance from ancestors, has been known to occur among prokaryotes for many years (Davies 1996) and is increasingly of interest in eukaryotes as well (Doolittle 1998; Hotopp et al. 2007). Horizontally acquired genes can affect how we develop and interpret sequence and functional annotation. The extent and sources of horizontal gene transfer (HGT) in an organism may even affect our ability to reconstruct the entire organism's evolutionary history (Doolittle 1999). A wide range of algorithms for identifying horizontal gene transfer have been suggested, from sequence composition methods to homology searching to phylogenetic approaches.

Sequence composition methods (Mrazek and Karlin 1999) rely on the observation that sequences transferred from a distant genome retain some of the codon and sequence bias of the original organism, which they lose over time (Lawrence and Ochman 1997). These are among the most efficient and scalable approaches to HGT detection, but they can fail in two important cases: when the transfers are ancient or when they are among sufficiently similar species.

I. Măndoiu, R. Sunderraman, and A. Zelikovsky (Eds.): ISBRA 2008, LNBI 4983, pp. 26–37, 2008.

Homology methods, in which conclusions are drawn from the species distributions of the genes' closest neighbors, also scale to whole genomes (Lander et al. 2001; Podell and Gaasterland 2007), but annotation errors, incomplete databases, and gene loss raise serious questions about the accuracy of such methods (Salzberg et al. 2001).

In many cases, the best-performing algorithms use phylogenetic approaches to reconstruct the evolutionary histories of genomes and individual genes (Eisen 2000). A number of such "tree-based" approaches have been considered, most of which compare inferred trees for individual genes to a "correct" tree showing the overall phylogenetic relationships of the considered species (Robinson 1981; Shimodaira 2002). Such methods are the only ones that incorporate putative evolutionary relationships. Bottom-up tree construction methods, such as the neighbor-joining algorithm (Saitou and Nei 1987), often identify fine structure successfully and so perform relatively well at identifying transfers even between similar species.

However, even tree-based approaches are imperfect. First, they generally require construction of a phylogenetic tree for each gene under consideration. Thus, they are slow and tend to scale poorly to genome-wide applications. In addition, inference of correct phylogenetic trees is a difficult problem, and inferred gene trees can be incorrect, particularly when lineages evolve at different rates (Anderson and Swofford 2004). There are two commonly-used approaches to building the "consensus" tree needed by typical tree-based methods: inferring a phylogenetic tree for each gene and then constructing a consensus of these trees; or else concatenating all the gene sequences together and inferring a tree representing these concatenated sequences. In either case, an incorrect consensus tree will cause errors in the entire algorithm.

We introduce an approach that has many of the strengths of phylogenetic approaches but avoids some of their pitfalls. Specifically, we use the same pairwise distances used by phylogenetic inference algorithms to detect horizontal transfer without building the trees themselves. Since determining the optimal tree topology is the most computationally-intensive part of the tree-based HGT-detection process, a distance-based approach runs much more quickly, allowing scanning of whole genomes. Furthermore, there is no "consensus" tree, so this method doesn't suffer in cases where no single tree that fits all of the data well. Instead, we consider how the *relative* pairwise distances between species in one gene family relate to those relative distances in another. Thus, our method can accommodate genes with different rates of evolution and genes that appear in different sets of species. Because it relies only on the pairwise species distances, we refer to this as the Distance Method.

As an example, consider the tree in Figure 1 showing five species (labeled A – E). One might expect that for most genes, sequence-derived distances between orthologs in D and E would be small, while distances between orthologs in A and D would be larger. However, suppose that a gene from D had relatively recently transferred into A's genome. Then the sequence-derived distance between that gene in D and its closest ortholog in A would be surprisingly small, while the distance between the orthologs in A and B would be surprisingly large. We detect these unusual events using these distances, avoiding the hazards of errors that can be introduced in the tree-construction process and the computational cost of building the trees. Our method compares the pairwise species distances among different gene families and reports the number of unusual-looking distances detected in each family.

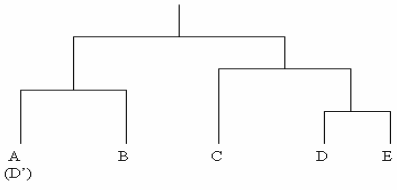


Fig. 1. A hypothetical tree of five species, A-E. Note that if a gene from D had been transferred into A's genome, the distances between that gene (D') in A and E would be surprisingly small, while that between A and B would be surprisingly large. Our method detects HGT by observing these differences.

2 Methods

This section first introduces the computational method used to identify horizontally transferred genes. Next, it describes the construction of test data sets used to evaluate the computational approach.

2.1 Identification of Horizontally Transferred Genes

Given a species in which we want to find HGT, we start by identifying a set of related species for comparison. In the experiments described here, we followed the example of (Lerat et al. 2003) in selecting *E. coli K12* as our target genome, and a commonly-studied set of 12 other gamma-Proteobacteria (see Table 1) as additional species for comparison. We aim to identify genes from *E. coli* whose evolutionary history *with respect to the other species in our data set* is unusual.

The basic assumption behind our algorithm is that, for a given pair of species, the sequence-derived distances between any two orthologous genes in those two species should be similarly *ranked*, when compared to the distances between other members of the same gene family in other species pairs. Note that this assumption allows for variation in the evolutionary rates of genes.

For example, in Figure 1, for any gene with orthologs in all five species we expect the corresponding sequences in species A and B to be closer than those in A and D. If they are not, it suggests that the evolutionary history of that gene may be atypical. Specifically, if a gene has been recently transferred from another species (whether among those in the data set or outside it), we expect these distance ranks to be unusual for *many* species pairs. Our algorithm identifies such genes. We refer to these HGT candidates as "outlier" genes because of their unusual distance distributions.

Computing Pairwise Species Distances. For each gene in the target genome, we identify all orthologous genes in each other species in the data set using BLASTP (Altschul et al. 1997; Schaffer et al. 2001) with an E-value below 10^{-20}. For simplicity, we use the single best BLAST hit in each species to identify orthologous genes, though ultimately more sophisticated approaches may be valuable (Remm et al. 2001; Podell and Gaasterland 2007). For each gene having at least three detectable orthologs

in species other than the target genome, we then construct multiple sequence alignments for the gene family using ClustalW (Thompson et al. 1994). Given these alignments, the protdist function in PHYLIP (Felsenstein 2002) calculates the distances between each pair of sequences in the alignment.

We note that multiple sequence alignments can be unreliable, just as inferred phylogenetic trees can, so the distances produced by PHYLIP may be incorrect. However, our method does not need to resolve inconsistencies among the distances by choosing a single tree. Thus, it may be less sensitive than phylogenetic methods to errors or inconsistencies in the inferred distances.

Detecting "Outlier" Genes. Different genes evolve at different rates. If we were to rely on raw distances to identify genes whose evolutionary history appears unusual, genes evolving particularly quickly or slowly would be at the top of the list. To avoid this effect, we first normalize the distance data. Specifically, for each gene family and species, we z-score normalize the set of pairwise distances between the gene in that species and all other species (in which a unique best ortholog for that gene is detectable). For example, in a data set of 190 genes in 13 species (as in Dataset 1, below), we would z-score normalize 2470 sets (corresponding to 190 genes * 13 species) of 12 distances each (corresponding to the other orthologs of that gene).

To pick our "outliers" we create a distance vector for each pair of species; in the same example, there would be 13 * 12 distance vectors, each of length 190 (corresponding to the total number of *E. coli* genes). For each pair of species, we then compute the mean and standard deviation of the values in *that distance vector*, and we identify as outliers any genes that are more than c standard deviations from the mean.[1] Then, we count up how many of these flagged outliers over all species-pair vectors belong to the same gene. A gene that is flagged as an outlier in this way in more than half the species pairs that include species S is considered an *outlier gene* for species S. We then consider species S an *outlier species* for that gene. Genes with one or more outlier species are reported as having an unusual evolutionary history.

Clustering Gene Families by Species. Though normalization is necessary to account for different rates of evolution, an unwelcome side-effect of normalization is that genes existing in only a small number of species are more likely to be chosen as outliers. However, to be applicable on a genome-wide level, our approach must be able to handle genes with detectable orthologs in only a small number of species. This missing-data bias disappears when comparisons are made among sets of genes occurring in roughly similar sets of species. Thus, we pre-process our data by clustering the *E. coli* genes according to the sets of species in which unique orthologs are identifiable. We call this procedure the *Hamming Distance Clustering* step.

To start, we define a *species set* as a set of genes whose orthologs are detectable (by the BLAST method described above) in exactly the same subset of the considered species. We call a species set *large* if it contains more than 30 genes, and we assume

[1] For all experiments in this paper, we choose $c = 2.326$, which would correspond to about 2% of the data in each vector if the distances are normally distributed. In practice, they are not, but the top half of the data – that part not constrained by the fact that distances must be non-negative – is close.

there are k species in our data set. Initially, each large species set becomes the core of its own cluster. We now extend these clusters to include the rest of the genes. We do this by an iterative process.

First, for each existing cluster C in decreasing order by size, let v_C be a binary vector of length k indicating the species in which the genes in C appear. Now, consider in turn each species set S not already clustered, and create binary indicator vector v_S for set S. If the Hamming distance between v_C and v_S is at most 2, merge S into cluster C (without changing v_C). When all S have been considered, we move on to the next core cluster and repeat the process. Finally, any remaining species sets are assigned to the cluster with the closest core Hamming distance. Once this pre-processing step has been completed, we run our outlier detection algorithm on each cluster and report any genes flagged as outliers in any cluster.

2.2 Construction of Test Data Sets

Here we describe the data sets we used to evaluate our approach. We started by downloading thirteen completed gamma-Proteobacteria genomes from the NCBI Genome database in November, 2006. Only the encoded protein sequences were used in this project. Table 1 summarizes the data from these 13 species, which are exactly those chosen by (Lerat et al. 2003). We then constructed three different data sets, which are summarized in Table 2.

Table 1. The thirteen gamma-Proteobacterial genomes from which our test data sets were constructed

Species	Abbrev.	Genome ID	# of proteins
Buchnera aphidicola APS	BA	NC_002528	564
Escherichia coli K12	EC	NC_000913	4,243
Haemophilus influenzae rd	HI	NC_000907	1,657
Pseudomonas aeruginosa PAO1	PA	NC_002516	5,566
Pasteurella multocida Pm70	PM	NC_002663	2,015
Salmonella typhimurium LT2	ST	NC_003197	4,425
Vibrio cholerae	VC	NC_002505,NC_002506	3,835
Wigglesworthia brevipalpis	WB	NC_004344	611
Xanthomonas axonopodis	XA	NC_003919	4,312
Xanthomonas campestris	XC	NC_003902	4,181
Xylella fastidiosa 9a5c	XF	NC_002488	2,766
Yersinia pestis CO92	YC	NC_003143	3,885
Yersinia pestis KIM	YK	NC_004088	4,086

Dataset 1: Comparison with Lerat's HGT Method. The first dataset was designed to determine whether we could identify the same cases of horizontal gene transfer (bioB and mivN) as the consensus-tree approach described in (Lerat et al. 2003). Of the 205 genes in their data set, we were able to identify 189 of them in our database (presumably because the *E. coli* genome annotation has changed somewhat in the intervening years). In fact, only 168 of the 189 genes had orthologs detected by our criteria in all of the 13 species considered. (This is because of differences between their methods and ours for identifying orthologs.) Given these differences, we therefore

added in one other known example of horizontal gene transfer, the tadA gene (Planet 2006). In total, Dataset 1 contains 190 genes.

Dataset 2: Calculating Sensitivity. This dataset is designed to test the sensitivity of our method. The problem with calculating the sensitivity, specificity, or indeed any measure of accuracy of an HGT detection method is that, for most real data, the right answers are unknown. Specifically, it is impossible to identify the line between true positives and false positives. However, we can take advantage of an idea of Poptsova and Gogarten (Poptsova and Gogarten 2007) to create a small subset of data where we know that the evolutionary history of some specific genes is abnormal, because we've "spiked" in those abnormal sequences ourselves.

In this data set, we restricted our attention to genes that were best reciprocal BLAST hits between each pair of species. Thus, only 148 genes from Dataset 1 were selected to form Dataset 2. To simulate horizontal gene transfer between *E. coli* and another species in the data set, we randomly select one of the other species and swap the orthologous gene sequences between *E. coli* and that other species.

In fact, Dataset 2 is really comprised of 10 sub-datasets. Each sub-dataset contains the same 148 genes, but includes 10 different randomly-chosen swapped genes. In total, there are 100 simulated "outlier" genes planted in Dataset 2.

Dataset 3: A Genomic-Scale Test Case. There are 4243 *E. coli K12* genes in the genome sequence we downloaded. However, for our distance method to work, we require that genes have more than 3 detectable orthologs among the 13 species. We selected all 2853 *E. coli* genes meeting this criterion, and their orthologs in the other species, to form Dataset 3.

Table 2. Summary of test data sets

	Dataset 1	**Dataset 2**	**Dataset 3**
Number of genes	190	148 per sub-dataset	2853
Number of subsets	None	10	None
Known outliers	None	10 per sub-dataset	None
Demonstrates	Feasibility	Sensitivity	Genomic Scale

3 Results

3.1 Feasibility

We first ran our algorithm on Dataset 1. The distance method identifies 19 of the 190 genes (10%) as outliers; these are listed in Table 3. Experts disagree on the expected prevalence of horizontal gene transfer in bacterial genomes (Martin 1999), but values between 5 and 15% of the genome are common, so identifying 10% of the input genes in this set seems reasonable. However, because this data set contains only widely-conserved genes, we do not necessarily expect this 10% outlier-detection rate to extend to the whole genome (see Section 3.3).

The 19 gene list includes all three known examples of HGT: tadA (Planet 2006) and mviN and bioB (Lerat et al. 2003). We also note that there are several ribosomal

proteins on the list; previous work suggests that horizontal gene transfer is common among ribosomal protein families (Coenye and Vandamme 2005).

Finally, the gene ileS appears on this list because of a database error: the *H. influenzae* genome sequence listed in Table 1 lacks the ileS (isoleucyl-tRNA synthetase) gene entirely. This is presumably a database error in the NCBI sequence – the gene itself is essential, and the gene appears in other versions of the genome. However, because of this absence, the best BLAST hit of the *E. coli* ileS gene in *H. influenzae* turns out to be valyl-tRNA synthetase. Thus, the evolutionary history of the gene appears to the algorithm to be unusual, so this gene is flagged as an outlier. We chose not to correct this error because its presence testifies to the algorithm's efficacy.

Table 3. Genes identified as outliers in Dataset 1. Known examples of HGT and the detected database error are shaded.

rank	# outlier species	# orthologs	Locus	Product name
1	5	5	secE	Translocase
2	4	13	ileS	isoleucyl-tRNA synthetase
2	4	4	rpmD	50S ribosomal protein L30
2	4	4	rpmF	50S ribosomal protein L32
2	4	7	rplO	50S ribosomal protein L15
6	2	13	bioB	Biotin synthase
6	2	13	mviN	Predicted inner membrane protein
6	2	13	ftsZ	cell division protein FtsZ
6	2	9	rplL	50S ribosomal protein L7/L12
6	2	7	rpmG	50S ribosomal protein L33
11	1	13	tadA	tRNA-specific adenosine deaminase
11	1	13	atpD	ATP synthase subunit B
11	1	13	ftsA	cell division protein
11	1	13	gltX	glutamyl-tRNA synthetase
11	1	13	htpX	heat shock protein HtpX
11	1	13	ribA	GTP cyclohydrolase II protein
11	1	8	rplY	50S ribosomal protein L25
11	1	13	rpsJ	30S ribosomal protein S10
11	1	11	yqgF	Holliday junction resolvase-like protein

3.2 Sensitivity

To evaluate the performance of the distance method, we used the simulated anomalies in Dataset 2. We combine the results from each of the ten trials to identify how many of 100 randomly "spiked" anomalous genes we were able to detect. For comparison, we also applied the AU ("approximately unbiased") test (Shimodaira 2002) to the same data. The AU test is a tree-based method that has been shown to perform well in identifying horizontal gene transfer (Poptsova and Gogarten 2007).

Overall, our distance method did not do as well as the AU test in finding the swapped genes in this data. Only 46 of the 100 swapped genes were identified, compared to 74 under the AU test method. However, a closer analysis of which swaps were found by each method yields some interesting insights.

Figure 2a shows the 100 swapped genes identified by the species with which the *E. coli* representative was swapped. The distance method failed to identify any swaps

between *E. coli* and the two *Y. pestis* genomes (YC and YK), which are highly similar to *E. coli*. But it did well on identifying swaps from many other organisms.

The most interesting phenomenon illustrated in Dataset 2 is that the distance method identifies all exchanges between *E. coli* and *B. aphidicola* or *W. brevipalpis*, while the AU test results are much weaker for these genes. These two species are endosymbionts, which are evolving more rapidly than other species in the data set. Their evolutionary relationship to each other and to the rest of the species in the data set is unclear. Some phylogenetic methods suggest that they are closely related to each other (Lerat et al. 2003), but others disagree (van Ham et al. 1997; Spaulding and von Dohlen 1998; Moya et al. 2002). We suspect that for sequences from these species, the tree-based AU test fails because long branch attraction (Anderson and Swofford 2004) creates errors in the consensus tree. However, the distance method does not suffer from this problem at all.

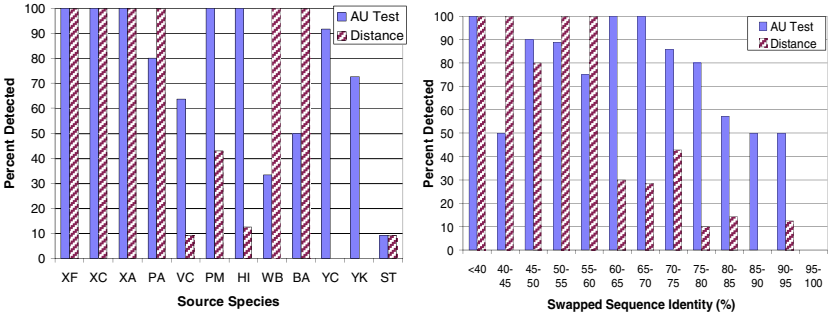


Fig. 2. Known outliers detected by the Distance Method and the AU test. a) Breakdown by source species. While the AU test outperforms the distance method overall, this is not the case for all species. In particular, the distance method identifies all spiked sequences deriving from the *B. aphidocola* and *W. brevipalpis* genomes. This is interesting because these symbiotic species are rapidly evolving, so many tree inference methods have trouble placing them correctly. The distance method for detecting outliers avoids this pitfall. b) Breakdown by percent identity of the swapped sequences. Distance outperforms the AU test for dissimilar sequences, but performance of the AU test falls off less dramatically as sequence similarity increases.

Figure 2b shows the Dataset 2 results broken down by the degree of sequence identity between the swapped genes. These results demonstrate that the distance method does well in identifying swapped sequences with only moderate sequence similiarity, even in cases where tree inference methods struggle, but it has trouble when the spiked sequences are too similar to those of the host genome.

It is possible that the distance method preferentially identifies rapidly-evolving genes, despite our attempts to account for this via normalization. To eliminate this possibility, we examined the “outlier species” for all 19 genes flagged as outliers in Dataset 1 (i.e., without swapped-in genes). Not a single gene was considered to be an outlier in *B. aphidicola* or *W. brevipalpis*. This is because the variance of the pairwise distances is large for these species, so we don’t identify their genes as being unusually

far from the mean. Thus, *z*-score normalization appears to be effective, and our ability to detect transfers with these species in Dataset 2 is not an artifact.

Finally, we measured the running times of the two methods on a typical run of Dataset 2 (one set of 148 genes). Both methods use BLAST and ClustalW as pre-processing steps, which took ~5.75 minutes on our 2.4 GHz linux machine. Building the trees in PAML and running the AU test code in CONSEL required 46.5 min., not including time needed to construct a consensus tree (already known for the 13 species involved). In contrast, the distance method required 2.5 min. to calculate pairwise distances in PHYLIP for all pairs of species in all gene families, and then a total of 0.41 seconds to identify outliers in all 148 genes.

3.3 A Genome-Scale Application

We ran the Distance Method on Dataset 3 to identify efficacy across a genomic-scale data set. A total of 214 genes (7.5%) were detected as outliers. The full list is available as supplementary data. Figure 3 shows that the probability of a gene's being detected as an outlier is slightly lower for genes detected in few species. This makes sense, because if the sequences exist in fewer species, there are fewer species pairs available to witness the unusual history for that gene. In addition, however, this result demonstrates that the clustering approach successfully overcomes any normalization-induced bias towards selecting genes that appear in few species.

The entire run took 110 minutes on our linux machine; 28 minutes of that was needed to compute distances in PHYLIP, and just under 6 seconds to identify outliers. In other words, the part of the distance method after the pre-processing step of identifying orthologs and aligning them (shared with the AU test) took less real time for the *entire genome* than the unique AU test calculations did for just 148 genes.

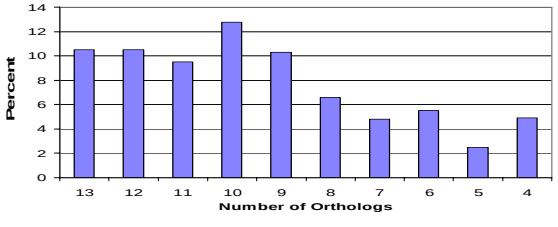


Fig. 3. Percentage of the 214 outliers from Dataset 3 with detectable orthologs in different numbers of species

To assess accuracy in this data set, we did not compare our results to tree-based methods, since none to our knowledge is suitable for genome-wide scanning. However, a newly-published method for re-ranking BLAST results has been proposed as a way to find previously-undetected HGT events on a genomic scale (Podell and Gaasterland 2007). We compare our results to theirs. In addition, we can search the literature for validation of our findings, though this is a labor-intensive process.

DarkHorse (Podell and Gaasterland 2007) identifies putative horizontal gene transfer using BLAST to detect closest neighbors, but extending attention beyond the single best BLAST hit. Their method has been shown to be applicable on a genomic

scale and more sensitive than traditional BLAST searching, and it has already been tested on the *E. coli* genome. We compared our results to those reported as supplemental data in the DarkHorse paper.

Because genes in Dataset 3 must have more than three detectable orthologs among the 13 species in our data set, many of the *E. coli* genes that DarkHorse predicts to be horizontally transferred are not included in Dataset 3. However, of the 2853 genes in Dataset 3, DarkHorse predicts that 31 of them are examples of HGT between *E. coli* and another species. Among our list of predicted outliers, we have only 7 in common with this list of 31: ygfK, ygfO, ydcU, yjhH, yjhG, yagE, and paaH.

This result raises two questions. First, how likely is it that we would find that many overlapping genes just by chance? To address this question, we chose 100 random sets of 214 genes from Dataset 3, and measured their intersection with the 31 genes in the DarkHorse list. In none of those 100 cases did we ever see seven intersecting genes, and in only one case did we even see as many as six.

Second, in the cases where the two methods disagree, which is correct? We offer no dispute of the DarkHorse predictions, except the general observation that different evolutionary rates, gene loss, and sequence annotation errors are known to limit the accuracy of homology-based methods (Eisen 2000). However, we manually searched for publications linking 60 of our predicted outliers to horizontal transfer between *E. coli* and another species. We found such evidence in 5 of the 60 cases: trkG (Ly et al. 2004), fhiA and fliS (Ren et al. 2005), agaV (Charbit and Autret 1998), and cmtA (Sprenger 1993). These data suggest that many of our novel predictions may be correct, and that a method that combines multiple approaches might be the best one.

4 Conclusions

Our results demonstrate the potential of using distances to detect HGT instead of full phylogenetic methods. The Distance Method described here identifies many known positive examples, including some missed by other methods, but also appears to miss some that other methods detect. Specifically, the Distance Method does particularly well at identifying outlier sequences with only moderate sequence similarity to the host gene, even in cases (such as rapidly evolving symbiotic organisms) where tree-inference methods often fail. On the other hand, the Distance Method struggles to detect transfers between closely related genomes. These transfers are challenging for any HGT method, but the AU Test outperforms the Distance Method here.

These results suggest that, if there were a fast (genomic-scale) tree-based method with accuracy similar to that of the AU Test, the best solution would be to combine that method with the Distance Method. We consider these initial results promising, and we expect that further development of such approaches will yield a scalable HGT-detection method with high accuracy and speed.

This work also has implications for another problem beyond that of HGT-detection: the detection of unnatural genes in the environment. Genetically-modified genomes may appear in the environment by accident, such as when genetically-modified organisms escape containment (Warwick et al. 2007), or by design, such as the malicious engineering of pathogenic organisms. We are interested in ways to identify signs of such “unnatural” DNA by sequence analysis. If we can reliably find

genes that appear to have been derived from a foreign source, content-based methods may help us infer whether the transfer was recent or ancient (Lawrence and Ochman 1997), and functional analysis may suggest whether the transfer occurred naturally or with human intervention. Thus, a distance-based approach to identifying atypical lineages may prove to be a powerful, scalable tool for finding unnatural DNA.

Acknowledgements

The authors gratefully acknowledge the support of AFOSR/DARPA seedling grant FA9550-06-1-0478. DS and XW were supported in part by NIH grant 1R21LM009411; AB and LC were supported in part by NSF grant (ASE+NHS)(dms) 0428715. Thanks to Jonathan Eisen and Sourav Chatterji for extremely valuable discussions.

References

Altschul, S.F., Madden, T.L., Schaffer, A.A., Zhang, J., Zhang, Z., Miller, W., Lipman, D.J.: Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. Nucleic. Acids. Res. 25(17), 3389–3402 (1997)

Anderson, F.E., Swofford, D.L.: Should we be worried about long-branch attraction in real data sets? Investigations using metazoan 18S rDNA. Mol Phylogenet Evol. 33(2), 440–451 (2004)

Charbit, A., Autret, N.: Horizontal transfer of chromosomal DNA between the marine bacterium Vibrio furnissii and Escherichia coli revealed by sequence analysis. Microb. Comp. Genomics 3(2), 119–132 (1998)

Coenye, T., Vandamme, P.: Organisation of the S10, spc and alpha ribosomal protein gene clusters in prokaryotic genomes. FEMS Microbiol. Lett. 242(1), 117–126 (2005)

Davies, J.: Origins and evolution of antibiotic resistance. Microbiologia 12(1), 9–16 (1996)

Doolittle, W.F.: You are what you eat: a gene transfer ratchet could account for bacterial genes in eukaryotic nuclear genomes. Trends Genet 14(8), 307–311 (1998)

Doolittle, W.F.: Lateral genomics. Trends. Cell. Biol. 9(12), 5–8 (1999)

Eisen, J.A.: Horizontal gene transfer among microbial genomes: new insights from complete genome analysis. Curr. Opin. Genet. Dev. 10(6), 606–611 (2000)

Felsenstein, J.: PHYLIP (Phylogeny Inference Package), version 3.66. Department of Genetics, University of Washington, Seattle, Washington (2002)

Hotopp, J.C., Clark, M.E., Oliveira, D.C., Foster, J.M., Fischer, P., Torres, M.C., Giebel, J.D., Kumar, N., Ishmael, N., Wang, S., et al.: Widespread lateral gene transfer from intra-cellular bacteria to multicellular eukaryotes. Widespread lateral gene transfer from intra-cellular bacteria to multicellular eukaryotes 317(5845), 1753–1756 (2007)

Lander, E.S., Linton, L.M., Birren, B., Nusbaum, C., Zody, M.C., Baldwin, J., Devon, K., Dewar, K., Doyle, M., FitzHugh, W., et al.: Initial sequencing and analysis of the human genome. Nature 409(6822), 860–921 (2001)

Lawrence, J.G., Ochman, H.: Amelioration of bacterial genomes: rates of change and exchange. J. Mol. Evol. 44(4), 383–397 (1997)

Lerat, E., Daubin, V., Moran, N.A.: From gene trees to organismal phylogeny in prokaryotes: the case of the gamma-Proteobacteria. PLoS Biol. 1(1), 19 (2003)

Ly, A., Henderson, J., Lu, A., Culham, D.E., Wood, J.M.: Osmoregulatory systems of Escherichia coli: identification of betaine-carnitine-choline transporter family member BetU and distributions of betU and trkG among pathogenic and nonpathogenic isolates. J. Bacteriol. 186(2), 296–306 (2004)

Martin, W.: Mosaic bacterial chromosomes: a challenge en route to a tree of genomes. Bioessays 21(2), 99–104 (1999)

Moya, A., Latorre, A., Sabater-Munoz, B., Silva, F.J.: Comparative molecular evolution of primary (Buchnera) and secondary symbionts of aphids based on two protein-coding genes. J. Mol. Evol. 55(2), 127–137 (2002)

Mrazek, J., Karlin, S.: Detecting alien genes in bacterial genomes. Ann. N. Y. Acad. Sci. 870, 314–329 (1999)

Planet, P.J.: Tree disagreement: measuring and testing incongruence in phylogenies. J. Biomed. Inform. 39(1), 86–102 (2006)

Podell, S., Gaasterland, T.: DarkHorse: a method for genome-wide prediction of horizontal gene transfer. Genome Biol 8(2), 16 (2007)

Poptsova, M.S., Gogarten, J.P.: The power of phylogenetic approaches to detect horizontally transferred genes. BMC Evol. Biol. 7, 45 (2007)

Remm, M., Storm, C.E., Sonnhammer, E.L.: Automatic clustering of orthologs and in-paralogs from pairwise species comparisons. J. Mol. Biol. 314(5), 1041–1052 (2001)

Ren, C.P., Beatson, S.A., Parkhill, J., Pallen, M.J.: The Flag-2 locus, an ancestral gene cluster, is potentially associated with a novel flagellar system from Escherichia coli. J. Bacteriol. 187(4), 1430–1440 (2005)

Robinson, D.: Comparison of phylogenetic trees. Mathematical Biosciences 53, 131–147 (1981)

Saitou, N., Nei, M.: The neighbor-joining method: a new method for reconstructing phylogenetic trees. Mol. Biol. Evol. 4(4), 406–425 (1987)

Salzberg, S.L., White, O., Peterson, J., Eisen, J.A.: Microbial genes in the human genome: lateral transfer or gene loss? Science 292(5523), 1903–1906 (2001)

Schaffer, A.A., Aravind, L., Madden, T.L., Shavirin, S., Spouge, J.L., Wolf, Y.I., Koonin, E.V., Altschul, S.F.: Improving the accuracy of PSI-BLAST protein database searches with composition-based statistics and other refinements. Nucleic. Acids. Res. 29(14), 2994–3005 (2001)

Shimodaira, H.: An approximately unbiased test of phylogenetic tree selection. Syst. Biol. 51(3), 492–508 (2002)

Spaulding, A.W., von Dohlen, C.D.: Phylogenetic characterization and molecular evolution of bacterial endosymbionts in psyllids (Hemiptera: Sternorrhyncha). Mol. Biol. Evol. 15(11), 1506–1513 (1998)

Sprenger, G.A.: Two open reading frames adjacent to the Escherichia coli K-12 tran-sketolase (tkt) gene show high similarity to the mannitol phosphotransferase system en-zymes from Escherichia coli and various gram-positive bacteria. Biochim. Biophys. Acta. 1158(1), 103–106 (1993)

Thompson, J.D., Higgins, D.G., Gibson, T.J.: CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. Nucleic. Acids. Res. 22(22), 4673–4680 (1994)

van Ham, R.C., Moya, A., Latorre, A.: Putative evolutionary origin of plasmids carrying the genes involved in leucine biosynthesis in Buchnera aphidicola (endosymbiont of aphids). J. Bacteriol. 179(15), 4768–4777 (1997)

Warwick, S.I., Legere, A., Simard, M.J., James, T.: Do escaped transgenes persist in nature? The case of an herbicide resistance transgene in a weedy Brassica rapa population. Mol. Ecol. (2007)

An Approach for Determining Evolutionary Distance in Network-Based Phylogenetic Analysis

Tingting Zhou[1,2], Keith C.C. Chan[2], Yi Pan[3], and Zhenghua Wang[1]

[1] National Laboratory for Paralleling and Distributed Processing, School of Computer, National University of Defense Technology, Changsha, Hunan, 410073, P.R. of China
[2] Department of computing, The Hong Kong Polytechnic University, Hong Kong, China
[3] Department of Computer Science, Georgia State University, 34 Peachtree Street, Suite 1450, Atlanta, GA 30302-4110, USA
ttyuren@gmail.com

Abstract. Network-based phylogenetic analysis explores phylogenetic relationships among different organisms by comparing their biological networks, especially metabolic networks. The differences between networks, often expressed as evolutionary distances, are normally measured using the plain *Jaccard distance*. In this paper, we show enzymes are different in phylogenetic conservation and topological importance, which are correlated significantly. Inspired by this observation, we propose a new approach to determine evolutionary distances. Our approach considers not only the number of different enzymes in different organisms, but also the phylogenetic or topological difference of individual enzymes. The resulting evolutionary distance measures are compared with the plain *Jaccard distance* by use of 16s rRNA-based distance as reference. It shows that new distance measures make errors smaller in all test cases of comparison.

Keywords: network-based phylogenetic analysis, metabolic network comparison, evolutionary distance, Jaccard distance, phylogenetic conservation, topological importance.

1 Introduction

One goal of phylogenetic analysis is to explore the cross-species natural connections or evolutionary history by exploring the difference among species[1]. Most previous researches are based on sequence alignment, in which single genes, proteins, especially small-subunit ribosomal RNAs(e.g. 16s or 18s rRNA), are often considered as the phylogenetic marks. In this case, evolutionary distances among organisms are always given by the difference of these corresponding molecular sequences. However, such sequence-based approaches would be influenced by horizontal gene transfer (HGT)[2], especially in bacteria and some unicellular eukaryotes. Unlike the sequence-based methods, network-based phylogenetic analysis is carried out by comparing different organisms' homogeneous networks. In such cases, evolutionary distance is often defined as the difference of organisms' biological networks.

I. Măndoiu, R. Sunderraman, and A. Zelikovsky (Eds.): ISBRA 2008, LNBI 4983, pp. 38–49, 2008.

Metabolic networks are hierarchical integration of metabolites, enzymes, reactions and the relationships among them. As the execution level of life, metabolic networks are known more exactly and explicitly than other biological networks (such as transcriptional regulatory networks, protein-protein interaction networks as well as signal transduction networks)[3]. Therefore they are used more often in the network-based phylogenetic analysis.

Comparing metabolic networks, however, is not an easy task. Composed of tens or even hundreds of interlaced pathways, global metabolic networks are very large and complicated, which could hardly be compared element by element. They are often considered as sets of nodes or edges so that evolutionary distance can be defined as the difference between the corresponding sets. Among all available difference measures, *Jaccard distance*(*JC*) is used most commonly[4-7]. Liao et al[5] has ever regarded metabolic networks as pathway sets, to determine distance by comparing the numbers of pathways present or absent in the organisms. However, enzyme sets are considered more proper than pathway sets when denoting metabolic networks, because enzymes are related to genome directly and their information is more local than pathways. Besides, the similar doubt lies in the work of Tohsato[7], in which reaction sets are concerned.

More recent work on *JC* also includes that of Ma and Zeng[6]. They constructed phylogenetic trees with evolutionary distance determined by three indices, *Jaccard index*, *Simpson index* and *Korbel index*, and showed *Jaccard index* performed well in phylogenetic analysis by comparison.

One of the most detailed phylogenetic analysis using set theory was carried out by Forst et al[4]. They used 'clean' metabolite-reaction bipartite graph to represent metabolic network but still regarded the plain *JC* as the evolutionary distance measure. Despite the intricate graph representation, only reactions were compared as the elements of set. Moreover, less enzymatic information was considered.

Although *JC* performed well in the previous researches of network-based phylogenetic analysis, it's hard to say that *JC* is adequate since elements are often different even in the same set. For example, both phylogeny[8] and topology[9,10] make enzymes different in the case of enzyme network.

In this paper, we show that enzymes are different in phylogenetic conservation and topological importance, which are correlated significantly in the case of directed enzyme networks. This observation inspires us to integrate phylogenetic conservation and topological importance of enzymes into the determination of evolutionary distance. Regarding them as some weights to *JC*, we propose a new approach to determine the evolutionary distance and obtain four new distance measures. Using the 16s rRNA-based distance[11] as reference, we compare these distances with the plain *JC*. Results show that in network-based phylogenetic analysis, evolutionary distance is decided not only by the number of enzymes present or absent, but also by their phylogenetic conservation and topological importance.

This contribution is organized as following. Section 2 is arranged to display the proposed approach in detail. Section 3 displays the results, which show the good performance of the evolutionary distance measures determined by our approach. The last section summarizes the conclusion.

2 An Approach for Determining Evolutionary Distance

In this section, with the explanation of plain *JC* and the consideration of individual enzymes' phylogenetic conservation and topological importance, we give the new approach in a unified way. After that we describe the definition and calculation of four new distances derived from this approach, namely *phylogenetic-extent-weighted Jaccard distance*, *degree-centrality-weighted Jaccard distance*, *closeness-centrality-weighted Jaccard distance* and *betweenness-centrality-weighted Jaccard distance* (JC_p, JC_d, JC_c, and JC_b respectively).

2.1 Definition of *Jaccard Distance*

Jaccard distance (*JC*) is a common measure of the set difference, which is often regarded as the evolutionary distance in the network-based phylogenetic analysis by use of the set theory[4-6,10,12]. Suppose A and B are two sample sets, $A\backslash\backslash B$ stands for their difference set an $A \cup B$ for their union set, then *JC* of A and B is defined as the proportion of the size of their difference set to that of their union set, as following:

$$JC(A,B) = \frac{|A \backslash\backslash B|}{|A \cup B|} = \frac{\sum_{e \in A\backslash\backslash B} 1}{\sum_{e \in A \cup B} 1} \tag{1}$$

where '$|\bullet|$' means to get the size of the set inside, or say the cardinality of the set.

In the case of enzyme graph comparison, *JC* of two enzyme graphs equals to the proportion of the number of enzymes appearing only once in both of graphs to the total number of enzymes they contain.

2.2 The Approach for Determining the Evolutionary Distance

As a matter of fact, metabolic enzymes have phylogenetic extent of their own[8], which means they have their preferential organisms and should be treated discriminatingly. Besides, enzymes are also different for their topological importance[9,10], and the topological importance reflects individual enzyme's evolution[13,14]. For these reasons, as the evolutionary distance the plain *JC* isn't suitable any more. Thus, taking the phylogenetic conservation and topological importance information of individual enzymes into account, we propose a new approach to determine the global phylogeny of organism, which is based on *JC* but in a weighted way.

Considering two different organisms A and B, the evolutionary distance between them is denoted as $Dis(A, B)$. Enzyme's phylogenetic extent(P) is used to measure its conservation, and the three centralities, degree centrality(C_d), closeness centrality(C_c) and betweenness centrality(C_b), are used to measure enzyme's topological importance from different points of view. All of them are regarded as weights of individual enzymes. Therefore, by use of these weights, we obtain four new evolutionary distances, JC_p, JC_d, JC_c, and JC_b correspondingly.

The definition of these evolutionary distances can be described in a unified way. Let $A \backslash\backslash B$ stand for A and B's difference set and $A \cup B$ for their union set, also let $w(e)$ denote some kind of weight of enzyme e, then the evolutionary distance can be defined as:

$$Dis(A,B) = \frac{\sum_{e \in A \backslash\backslash B} w(e) \cdot 1}{\sum_{e \in A \cup B} w(e) \cdot 1} \tag{2}$$

where $Dis \in \{JC, JC_p, JC_d, JC_c, JC_b\}$ and $w \in \{P, C_d, C_c, C_b\}$. It is the ratio of weighted enzyme number in A and B's difference set to that in their union set. When all of the weights are set to 1, $Dis(A, B)$ equals to the plain JC.

2.3 Calculation of Weights

As described in the previous section, the key of the new approach lies in how to determine the conservation and topological importance of enzymes, which correspond to the four weights, *phylogenetic extent (P)*, *degree centrality* (C_d), *closeness centrality* (C_c), and *betweenness centrality* (C_b).

Phylogenetic Conservation Weight. Enzymes are proteins that catalyze metabolic reactions vital for the survival and functioning of cells. They tend to be either conserved or eliminated in the tree of life along with the evolution. This property is often expressed as phylogenetic profile[12], a binary string which encodes the presence ('1') or absence ('0') of an enzyme in every species. Following the example of Liu et al[9], we use *phylogenetic extent*(P) to measure the phylogenetic conservation of enzymes, which is defined as the sum of bits in phylogenetic profile, or say the number of organisms that contain the certain enzyme.

In brief, P equals to the number of '1' in the profile string. Given the total number of investigated species, P is proportional to the ratio of species that contain the certain enzyme. It shows how conservative the enzyme is to some extent.

Topological Importance Weights. In this work, metabolic network is represented as directed enzyme graph, and three common centralities are used to describe the topological importance of nodes from different points of view. They are the node's connectivity (*degree centrality*), its shortest paths to other nodes(*closeness centrality*) and the number of shortest paths going through the node(*betweenness centrality*) respectively. A node with high degree centrality may be important because of its many direct connections with others in the same work. A node with low closeness centrality would also be important since its influence could reach others in a short time. Nodes with high betweenness centrality would be important either, since they mediate many interactions between other nodes.

Degree centrality. As a common centrality measure, degree centrality describes a node's importance by counting the number of its direct interactions[15]. Formally, for

an undirected graph, the degree centrality of the node v is given by v's degree, which is also the total number of v's neighbors. For the directed network, degree centrality has two notions: one based on in-degree and the other based on out-degree, which are given by the number of edges that terminate and start at v respectively. Since either in-degree or out-degree belongs to the connection here, the degree centrality is given by the sum of them, that is:

$$C_d(v) = C_{din}(v) + C_{dout}(v). \tag{3}$$

where $C_d(v)$, $C_{din}(v)$ and $C_{dout}(v)$ are the degree centrality, in-degree and out-degree of node v respectively. In this case, whether the network is directed or undirected doesn't change the degree centrality.

Closeness centrality. Closeness centrality is used to describe how important a node is by measuring how 'close' it is to, or how quickly it can communicate with, the other nodes in the network [16]. In [9], it is defined as the sum of shortest distances from the given node to all of the others reachable in the same network. Nevertheless, it should be noticed that their enzyme graph was undirected. Considering the additional direction of edges could change the nodes' reachability, we define closeness centrality as the mean shortest path length between a vertex v and all other vertices reachable from it. To each node v in the network, its closeness centrality can be described as

$$C_c(v) = \frac{\sum_{t \in V \setminus v} d_G(v,t)}{n-1} \tag{4}$$

where V denotes the set of nodes that are reachable to v. $n>2$ is the size of V, and $d_G(v,t)$ is the shortest path from v to t in the directed graph G.

Betweenness centrality. Betweenness centrality is used to quantify an individual's influence in a network. It illustrates how important a node is by measuring how high proportion of paths it will mediate between other nodes[16]. For a graph with n vertices, let σ_{st} be the total number of shortest paths between s and t, and $\sigma_{st}(v)$ be the number of shortest paths between s and t that pass through v, then the betweenness centrality of v, $C_b(v)$, is given by:

$$C_b(v) = \sum_{\substack{s \neq v \neq t \in V \\ s \neq t}} \frac{\sigma_{st}(v)}{\sigma_{st}} \tag{5}$$

Liu et al[9] adopted the similar definition as ours. But it should be noticed that since the additional direction of edges could change nodes' reachability, the betweenness centrality of node in directed networks is different than undirected ones.

In this paper, the degree centrality is obtained by counting neighbors of a node directly. The Floyd-Warshall algorithm is used to compute the shortest path lengths to

get the closeness centrality, and the fast algorithm proposed by Brandes[17] is modified to calculate nodes' betweenness centrality in directed graph.

3 Results and Discussion

Enzyme and reaction data are obtained from metabolic database in [18], which is based on the famous KEGG/LIGAND database, with reaction reversibility added and current metabolites[1] eliminated. In this paper, metabolic network is represented as direct enzyme graph, in which vertices denote individual enzymes and arcs denote the relationships between them. If there is not less than one metabolite listed in both of one enzyme's products and the other's substrates, there will be an arc directed from the first enzyme to the second. The bidirectional arc is replaced by two individual arcs with opposite direction. And the enzyme with incomplete EC number (such as *EC 6.*-.-.-) is considered as an individual enzyme normally for its functional specialty.

Firstly we construct a global metabolic network for all the 107 organisms (8 Eukaryotes, 83 Bacteria and 16 Archaea), 1271 nodes and 10,029 arcs in total. In order to illustrate the four weights' reasonability, the phylogenetic conservation is studied for all the enzymes in the network (section 3.1). Then the correlation coefficient between phylogenetic conservation and topological importance is calculated, to show their significant correlation(section 3.2).

In section 3.3, for the sake of comparison, we choose 73 organisms (62 Bacteria and 11 Archaea) which are involved by both [11] and [18], and construct another enzyme graph for these 73 organisms in the same way, which contains 988 nodes and 7132 arcs. The weights are calculated respectively, and then the four evolutionary distances are determined by the proposed approach. After that these new distances are compared with *JC* respectively by using 16s rRNA-based distance as reference, which is given by Zhang et al[11].

3.1 Phylogenetic Conservation of Enzymes

The phylogenetic profile of enzymes can be obtained for the 107 organisms in the database easily. Their phylogenetic extent ranges from 1 to 104. A total of 158 enzymes appear in only one organism and 8 enzymes appear for 104 times within the 107 organisms. No one appears in all the 107 organisms. Most enzymes lie in few organisms, showing enzymes exhibit organism-specificity. And few enzymes appear in most organisms, showing they also exhibit some kind of conservation. It also reveals the phylogenetic conservation of enzymes is relative and various. In addition, we query the 10 enzymes with the topmost phylogenetic extent, and find 80% of them belong to transferring of phosphorus-containing groups, '*EC 2.7.*-.-' (GO:0016772), which indicates that the transferring of phosphorus-containing groups is one of the most conservative functions.

[1] It refers to ATP, ADP, NADH, NAD^+, H_2O, and so on, which is also known as redundant or external metabolites.

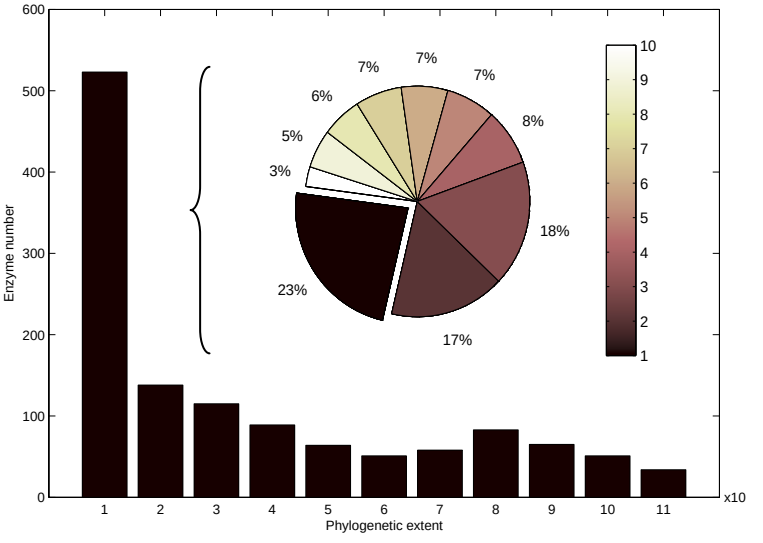


Fig. 1. Distribution of enzyme number along enzyme phylogenetic extent(*P*). The x-coordinate indicates increasing *P* from 1 to 107. The numbers under bars denote the range of *P*, e.g., '1' indicates the range of 0~10. The y-coordinate indicates the number of enzymes, whose *P* is in the corresponding range. The first range of *P* is also displayed as a pie chart to show the detailed ratio, and the ratio of $P = 1$ is split out.

3.2 Correlation between Conservation and Topological Importance

For the sake of further analysis, we compute the three centralities for each enzyme in the directed enzyme network of 107 organisms, namely *degree centrality* (C_d), *closeness centrality* (C_c) and *betweenness centrality* (C_b). Looking upon each enzyme as a sample, we test the hypothesis to determine the correlation between the three centralities and the profile. The correlation coefficients and the corresponding *p*-values are obtained with MATLAB, and shown in Table 1.

Table 1. The correlation between *P* and three centralities

	P -- C_d	$P - C_c$	$P - C_b$
Correlation coefficient	0.28	0.16	0.30
p-value	2.39E-24	0	9.86E-27

All the three *p*-values are very small, especially *P* and C_c's. It shows that significant correlation exists between phylogenetic conservation and topological importance of enzymes. Of all the three correlation coefficients, *P* and C_c's is the smallest and *P* and C_b's is the biggest, which means enzyme conservation correlates best with betweenness centrality but poorly with closeness centrality. It is consistent with the conclusion of Liu et al[9], although the network and the definition of centralities are

both different. Thus, the phylogenetic conservation and the topological importance are correlated significantly and then influence each other, no matter the network is directed or not.

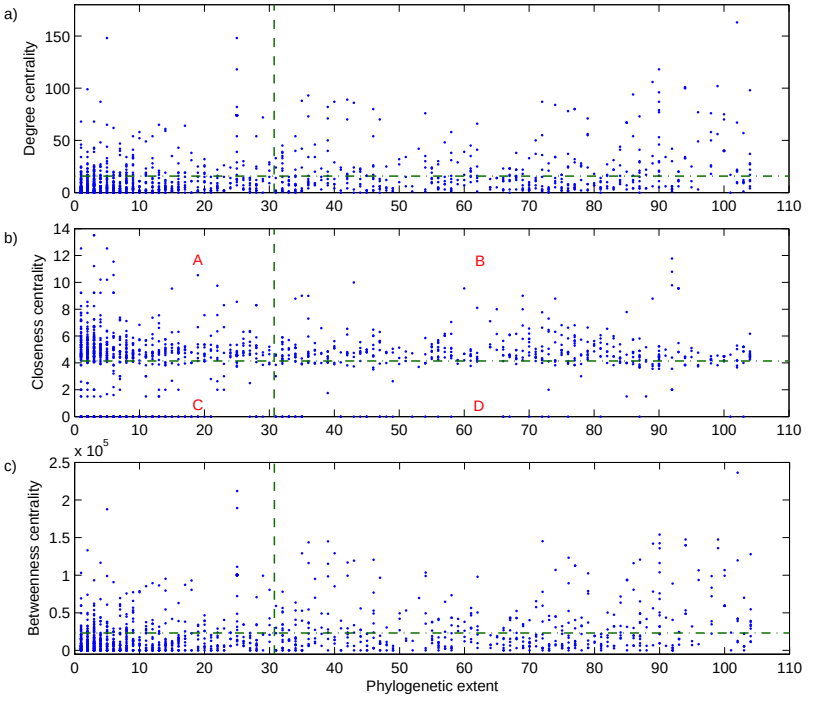


Fig. 2. Scatter figures of phylogenetic extent and three centralities. a) the correlation between phylogenetic extent and degree centrality; b) the correlation between phylogenetic extent and closeness centrality; c) the correlation between phylogenetic extent and the betweenness centrality. The horizontal broken indicates the mean of centrality and the vertical broken line indicates the mean phylogenetic extent.

Illustrated by the case of Fig 2.b), each subgraph can be divided into four regions by the horizontal and vertical broken lines which correspond to the mean values of centrality and phylogenetic extent respectively. We name the four regions A, B, C, and D. As is shown in the Fig 2.a) and the Fig 2.c), the region C is, to some extent, rather dense. While in the Fig 2.b), region A contains a large number of nodes. Since the nodes in regions B and C contribute to the positive correlation while the nodes in regions A and D destroy it, it explains why the correlation between P and C_d as well as P and C_b are higher than P and C_c.

3.3 Comparison of Evolutionary Distances

In this section, a global enzyme network for 73 organisms is constructed. Then P and C_d, C_c as well as C_b are calculated for each enzyme within it, and the corresponding

evolutionary distances(JC_p, JC_d, JC_c, and JC_b) are obtained for 72 pair of organisms (*eco* to other 72 organisms) by use of the proposed approach. Moreover, the plain *JC* is also obtained for the same organism pairs in the traditional way. For the sake of comparison, the 16s rRNA-based distance in [11] is regarded as reference. The result of comparison shows that the four evolutionary distances we defined are closer to the reference than the original *Jaccard distance* (Fig 3), which proves that either phylogenetic conservation or topological importance of individual enzymes contribute to the definition of evolutionary distance.

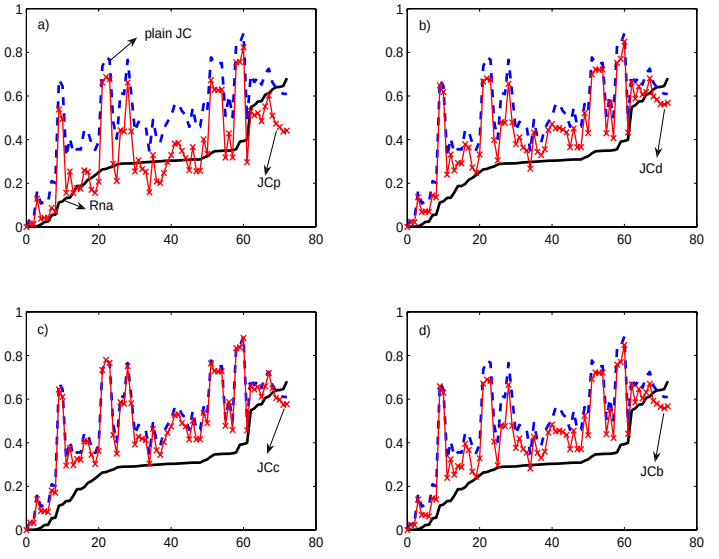


Fig. 3. Comparison between the proposed evolutionary distances (solid with stars) and the plain *JC* (dashed) with the RNA-based distance (solid) as reference. The four improved distances in four subgraphs are JC_p, JC_d ,JC_c and JC_b respectively. The x-coordinate denotes organisms, number corresponds to the organisms in sequence, sorted according to the distance value with *E.Coli K-12* as reference(see Appendix), which is denoted as the y-coordinate.

However, there are some big errors occurring around the x-coordinate 11, 24, 29, 54, 61 and so on. We think it may be caused by 'size bias', a drawback of the Jaccard index. That is, if the sizes (the number of enzymes in the metabolic network) of two sets are very different, the distance between them will be large, even if most of the enzymes in the smaller network are the same as that of the larger network[6]. Within the organisms we studied, *E coli K-12* has the largest network with 601 enzymes and *Ureaplasma urealyticum* has the smallest with only 72 enzymes. The large errors of the five distances are mostly recorded when an organisms has a size which is small enough.

Results of statistical hypothesis testing indicate all the six distances are highly correlated to each other (Table 2). For the sake of concision, the corresponding p-values are not shown here, all of which are no more than 3.2E-11. Of all the correlation coefficients, those between the five *JC*-based distances are higher than those between

them and the RNA-based. It means although the addition of enzymes' conservation or topological importance can reduce the error of the plain *JC*, it doesn't change the nature of set-theory-based distance definition. It also can be illustrated by Fig.3, in which the curves of the five *JC*-based distances look rather like each other in shape while they are all different than that of the RNA-based.

Table 2. The correlation between the six evolutionary distances

	RNA	*JC*	JC_p	JC_d	JC_c	JC_b
RNA	1	0.70	0.61	0.70	0.68	0.69
JC	0.70	1	0.97	0.99	1.00	0.99
JC_p	0.61	0.97	1	0.98	0.98	0.98
JC_d	0.70	0.99	0.98	1	0.99	1.00
JC_c	0.68	1.00	0.98	0.99	1	0.99
JC_b	0.69	0.99	0.98	1.00	0.99	1

We also compute and compare the errors of the five distances to the reference (Table 3). Compared with the error of *JC*, all the four weighted distances have smaller mean error. JC_p's mean error is smallest which is only 0.054, but its standard deviation is the biggest. Corresponding to Fig 3 a), the curve of JC_p is the one nearest to that of the reference. While JC_d's standard deviation is the smallest, although its mean error is the second smallest among the four weighted distance. It may indicate that the addition of between centrality makes *JC* insensitive to 'size bias' to some extent.

Table 3. Errors' mean and standard deviation

	Err_{JC}	Err_{JCp}	Err_{JCd}	Err_{JCc}	Err_{JCb}
Mean	0.2055	0.0540	0.1372	0.1759	0.1392
Standard deviation	0.1482	0.1662	0.1462	0.1536	0.1473

4 Conclusion

A lot of effort has been put into phylogenetic analysis by comparing metabolic networks. In this paper, we have made some initial attempts to integrate evolutionary information as well as topological importance of enzymes into the definition of evolutionary distance, which is shown with positive results. We not only show that in the case of directed enzyme graph the phylogenetic conservation and topological importance of enzymes are correlated significantly, but also illustrate that the conservation of individual enzymes contributes to the improvement of evolutionary distance definition and so does topological importance. It indicates that evolutionary distance is not only decided by the different coverage of enzymes, but also reflected by the conservation or topological importance of the present or absent enzymes. Thus, phylogenetic conservation as well as topological importance of individual enzymes should not be neglected in the network-based phylogenetic analysis.

Acknowledgments. We thank Dr. Hong-Wu Ma and Prof. Dr. An-Ping Zeng for sharing their revised metabolic database. We also thank the anonymous reviewers, whose comments have led to a substantial improvement of this article. This work is supported by the National Natural Science Foundation of China (60603054).

References

1. Hong, S.H., Kim, T.Y., Lee, S.Y.: Phylogenetic analysis based on genome-scale metabolic pathway reaction content. Applied Microbiology and Biotechnology 65(2), 203–210 (2004)
2. Wolfa, Y.I., Rogozina, I.B., Grishinb, N.V., Koonin, E.V.: Genome trees and the tree of life. Trends in Genetics 18(9), 472–479 (2002)
3. Sauer, U.: Metabolic networks in motion: 13 C-based flux analysis. Molecular Systems Biology 2(62) (2006)
4. Forst, C.V., Flamm, C., Hofacker, I.L., Stadler, P.F.: Algebraic comparison of metabolic networks, phylogenetic inference, and metabolic innovation. BMC Bioinformatics 7(1), 67–78 (2006)
5. Liao, L., Kim, S., Tomb, J.F.: Genome comparisons based on profiles of metabolic pathways. In: Proc. of the 6th International Conference on Knowledge-Based Intelligent Information and Engineering Systems, pp. 469–476 (2002)
6. Ma, H.W., Zeng, A.P.: Phylogenetic comparison of metabolic capacities of organisms at genome level. Molecular Phylogenetics and Evolution 31(1), 204–213 (2004)
7. Tohsato, Y.: A Method for Species Comparison of Metabolic Networks Using Reaction Profile. IPSJ Digital Courier 2(0), 685–690 (2006)
8. Peregrin-Alvarez, J.M., Tsoka, S., Ouzounis, C.A.: The Phylogenetic Extent of Metabolic Enzymes and Pathways. Genome Research 13(3), 422–427 (2003)
9. Liu, W., Lin, W., Davis, A., Jordan, F., Yang, H., Hwang, M.: A network perspective on the topological importance of enzymes and their phylogenetic conservation. BMC Bioinformatics 8(121) (2007)
10. Zhu, D., Qin, Z.S.: Structural comparison of metabolic networks in selected single cell organisms. BMC Bioinformatics 6(8) (2005)
11. Zhang, Y., Zhang, Z., Ling, L., Shi, B., Chen, R.: Conservation analysis of small RNA genes in Escherichia coli. Bioinformatics 20(5), 599–603 (2004)
12. Yamada, T., Kanehisa, M., Goto, S.: Extraction of phylogenetic network modules from the metabolic network. BMC Bioinformatics 7(1) (2006)
13. Lu, C., Zhang, Z., Leach, L., Kearsey, M.J., Luo, Z.W.: Impacts of yeast metabolic network structure on enzyme evolution. Genome Biology 8(407) (2007)
14. Vitkup, D., Kharchenko, P., Wagner, A.: Influence of metabolic network structure and function on enzyme evolution. Genome Biology 7(R39) (2006)
15. Aittokallio, T., Schwikowski, B.: Graph-based methods for analysing networks in cell biology. Briefings in Bioinformatics 7(3), 243 (2006)
16. Mason, O., Verwoerd, M.: Graph Theory and Networks in Biology. IET Syst. Biol. 1(2), 89–119 (2007)
17. Brandes, U.: A faster algorithm for betweenness centrality. Journal of Mathematical Sociology 25(2), 163–177 (2001)
18. Ma, H.W., Zeng, A.P.: Reconstruction of metabolic networks from genome data and analysis of their global structure for various organisms. Bioinformatics 19(2), 270–277 (2003)

Appendix: The List of 73 Organisms

X	O	S	X	O	S	X	O	S
1	eco	601	26	sco	491	50	hpy	278
2	ece	586	27	spy	265	51	smu	320
3	ecs	587	28	spg	266	52	tpa	145
4	ecc	510	29	cef	153	53	cpa	159
5	sfl	550	30	spm	269	54	cpn	160
6	sty	578	31	cac	404	55	cmu	158
7	stm	598	32	cgl	404	56	cte	331
8	ypk	514	33	lin	385	57	fnu	371
9	ype	538	34	lmo	397	58	tma	326
10	bas	195	35	bsu	493	59	mpu	105
11	buc	213	36	cpe	368	60	mpn	98
12	son	463	37	oih	460	61	uur	72
13	hin	365	38	mtc	444	62	aae	332
14	pae	543	39	syn	404	63	mja	262
15	xcc	482	40	mle	345	64	ape	273
16	xac	475	41	cje	308	65	mth	275
17	xfa	352	42	san	295	66	afu	313
18	nme	360	43	sag	292	67	mka	213
19	rso	571	44	spr	317	68	pho	225
20	sme	585	45	tte	339	69	tvo	283
21	ccr	468	46	sam	393	70	hal	299
22	bja	253	47	spn	306	71	sso	337
23	rpr	140	48	sav	396	72	mma	347
24	rco	150	49	sau	395	73	mac	333
25	tel	361						

Note: ‘X’ denotes the x-coordinates;
‘O’ denotes the abbreviation of organisms according to KEGG;
‘S’ stands for the size of enzyme set corresponding to each organism, which is also the enzyme number in the organism.

Pairwise Statistical Significance Versus Database Statistical Significance for Local Alignment of Protein Sequences

Ankit Agrawal[1], Volker Brendel[2], and Xiaoqiu Huang[1]

[1] Department of Computer Science, Iowa State University,
226 Atanasoff Hall, Ames, IA 50011-1041, USA
{ankitag,xqhuang}@iastate.edu
[2] Department of Genetics, Development, and Cell Biology and Department of Statistics, Iowa State University,
2112 Molecular Biology Building, Ames, IA, 50011-3260, USA
vbrendel@iastate.edu

Abstract. An important aspect of pairwise sequence comparison is assessing the statistical significance of the alignment. Most of the currently popular alignment programs report the statistical significance of an alignment in context of a database search. This database statistical significance is dependent on the database, and hence, the same alignment of a pair of sequences may be assessed different statistical significance values in different databases. In this paper, we explore the use of pairwise statistical significance, which is independent of any database, and can be useful in cases where we only have a pair of sequences and we want to comment on the relatedness of the sequences, independent of any database. We compared different methods and determined that censored maximum likelihood fitting the score distribution right of the peak is the most accurate method for estimating pairwise statistical significance. We evaluated this method in an experiment with a subset of CATH2.3, which had been previoulsy used by other authors as a benchmark data set for protein comparison. Comparison of results with database statistical significance reported by popular programs like SSEARCH and PSI-BLAST indicate that the results of pairwise statistical significance are comparable, indeed sometimes significantly better than those of database statistical significance (with SSEARCH). However, PSI-BLAST performs best, presumably due to its use of query-specific substitution matrices.

Keywords: Database statistical significance, Homologs, Pairwise local alignment, Pairwise statistical significance.

1 Introduction

Sequence alignment is extremely useful in the analysis of DNA and protein sequences [1,2,3]. Sequence alignment forms the basic step of making various high level inferences about the DNA and protein sequences - like homology, finding protein function, protein structure, deciphering evolutionary relationships, etc.

I. Măndoiu, R. Sunderraman, and A. Zelikovsky (Eds.): ISBRA 2008, LNBI 4983, pp. 50–61, 2008.

There are many programs that use some well known algorithms [4,5] or their heuristic version [3,6,7]. Recently, some enhancements in alignment program features have also become available [8,9] using difference blocks and multiple scoring matrices. Quality of a pairwise sequence alignment is gauged by the statistical significance rather than the alignment score alone, i.e., if an alignment score has a low probability of occurring by chance, the alignment is considered statistically significant.

For ungapped alignments, rigorous statistical theory for the alignment score distribution is available [10], and it was shown that the statistical parameters K and λ can be calculated analytically for a pair of sequences with given amino acid composition and scoring scheme. However, no perfect theory currently exists for gapped alignment score distribution, and for score distributions from alignment programs using additional features like difference blocks [8], and which use multiple parameter sets [9]. The problem of accurately determining the statistical significance of gapped sequence alignment has attracted a lot of attention in the recent years [11,12,13,14,15]. There exist a couple of good starting points for statistically describing gapped alignment score distributions [16,17], but a complete mathematical description of the optimal score distribution remains far from reach [17]. Some excellent reviews on statistical significance in sequence comparison are available in the literature [18,19,20].

Pairwise protein local sequence alignment programs give the optimal or suboptimal alignment of a given sequence pair. In the case of database searches, the second sequence is the complete database consisting of many sequences. Many approaches exist currently to estimate the statistical significance of a database hit (match of the query sequence with part of the database). For the database searches, the statistical significance of a pairwise alignment score is reported in terms of E-value, which is the expected number of hits in the database with a score equal or higher arising by chance, or the P-value, which is the probability of getting at least one score equal or higher arising by chance. These E-values and P-values are corresponding to the database, and although these can be converted to the pairwise E-values and P-values [15], they cannot estimate the true statistical significance of the specific pairwise alignment under consideration, since the database E-values and P-values depend on the average sequence features like length, amino acid composition, and not the features of sequence pair under consideration.

In particular, BLAST2.0 [3] reports the statistical significance as the likelihood that a similarity as good or better would be obtained by two random sequences with average amino-acid composition and lengths similar to the sequences that produced the score. However, if either of the two sequences has amino acid composition significantly different from the average, the statistical significance may be an over or underestimate. Similarly, the statistical estimates provided by the FASTA package [6,21] report the expectation that a sequence would obtain a similarity score against an unrelated sequence drawn at random from the sequence database that was searched, which again is dependent on the average sequence composition of the entire database and not on the specific sequence pair.

Accurate estimates of the statistical significance of pairwise alignments can be very useful to comment on the relatedness of a pair of sequences aligned by an alignment program independent of any database. And thus, pairwise statistical significance can also be used to compare different alignment programs independently. In addition to the standard local alignment programs [4,5], some recent programs have been developed [8,9] that take into account other desirable biological features in addition to gaps, like difference blocks, and the use of multiple parameter sets (substitution matrices, gap penalties). These features of the alignment programs enhance the sequence alignment of real sequences by suiting to different conservation rates at different spatial locations of the sequences. As pointed out earlier, rigorous statistical theory for alignment score distribution is available only for ungapped alignment, and not even for its simplest extension, i.e., alignment with gaps. Accurate statistics of the alignment score distribution from newer and more sophisticated alignment programs therefore is not expected to be straightforward. For comparing the performance of newer alignment programs, accurate estimates of pairwise statistical significance are needed.

The statistical significance of a pairwise alignment depends upon various factors: sequence alignment method, scoring scheme, sequence length, and sequence composition [19]. The straightforward way to estimate statistical significance of scores from an alignment program for which the statistical theory is unavailable is to generate a distribution of alignment scores using the program with randomly shuffled versions of the pair of sequences and compare the obtained score with the generated score distribution, either directly or by fitting an extreme value distribution (EVD) curve to the generated distribution to calculate the statistical significance of the obtained score (as described in the next section).

The PRSS program in the FASTA package [6,7,21] calculates the statistical significance of an alignment by aligning them, shuffling the second sequence up to 1000 times, and estimating the statistical significance from the distribution of shuffled alignment scores. It uses maximum likelihood to fit an EVD to the shuffled score distribution. A similar approach is also used in HMMER [22]. It also uses maximum likelihood fitting [23] and also allows for censoring of data left of a given cutoff, for fitting only the right tail of the histogram. A heuristic approximation of the gapped local alignment score distribution is also available [11], and based on these statistics, accurate formulae for statistical parameters K and λ for gapped alignments are derived and implemented in a program called ARIADNE [12]. These methods can provide an accurate estimation of statistical significance for gapped alignments, but currently do not incorporate the additional features of sequence alignment, like using difference blocks and multiple parameter sets [8,9].

The contribution of this paper is two-fold: First, we compare various existing methods to estimate pairwise statistical significance and determine the most accurate method for estimating it. We found that maximum likelihood fitting of score distribution censored left of peak (fitting right of peak) is the most accurate method. Secondly, we used this method in the experiments reported in [24]

on a subset of the CATH2.3 database to compare the retrieval accuracy for pairwise statistical significance and database statistical significance. [24] had earlier created this database to evaluate seven protein structure comparison methods and the two sequence comparison programs SSEARCH and PSI-BLAST. Comparison of the results with those reported in [24] show that pairwise statistical significance gives comparable and at times better accuracy than the SSEARCH program, but less than PSI-BLAST.

2 The Extreme Value Distribution for Ungapped and Gapped Alignments

Just as the distribution of the sum of a large number of independent identically distributed (i.i.d) random variables tends to a normal distribution (central limit theorem), the distribution of the maximum of a large number of i.i.d. random variables tends to an extreme value distribution (EVD). This is an important fact, because it allows us to fit an EVD to the score distribution from any local alignment program, and use it for estimating statistical significance of scores from that program. The distribution of Smith-Waterman local alignment score between random, unrelated sequences is approximately a Gumbel-type EVD [10]. In the limit of sufficiently large sequence lengths m and n, the statistics of HSP (High-scoring Segment Pairs which correspond to the ungapped local alignment) scores are characterized by two parameters, K and λ. The probability that the optimal local alignment score S exceeds x is given by the P-value:

$$\Pr(S > x) \sim 1 - e^{-E},$$

where E is the E-value and is given by

$$E = Kmne^{-\lambda x}.$$

For E-values less than 0.01, both E-value and P-values are very close to each other. The above formulae are valid for ungapped alignments [10], and the parameters K and λ can be computed analytically from the substitution scores and sequence compositions. An important point here is that this scheme allows for the use of only one substitution matrix. For the gapped alignment, no perfect statistical theory has yet been developed, although there exist some good starting points for the problem as mentioned before [16,17]. Recently, researchers have also looked closely at the low probability tail distribution, and the work in [25] applied a rare-event sampling technique and suggested a Gaussian correction to the Gumbel distribution to better describe the rare event tail, resulting in a considerable change in the reported significance values. However, for most practical purposes, the original Gumbel distribution has been widely used to describe gapped alignment score distribution [26,21,12,27,9].

From an empirically generated score distribution, we can directly observe the E-value E for a particular score x, by counting the number of times a score x or higher was attained. Since this number would be different for different number of

random shuffles N (or number of sequences in the database in case of database search), a normalized E-value is defined as

$$E_{normalized} = \frac{E}{N} \ .$$

3 Tools and Programs Used

We worked with the alignment programs SIM [28], which is an ordinary alignment program (similar to SSEARCH), GAP3 [8], which allows dynamically finding similarity blocks and difference blocks, and GAP4 [9], which can also use multiple parameter sets (scoring matrices, gap penalties, difference block penalties) to generate a single pairwise alignment. For estimating the statistical parameters K and λ, we used several programs. First is PRSS from the FASTA package [6,7,21], which takes two protein sequences and one set of parameters (scoring matrix, gap penalty), generates the optimal alignment, and estimates the K and λ parameters by aligning up to 1000 shuffled versions of the second sequence, and fitting an EVD using maximum likelihood. In addition to uniform shuffling, it also allows for windowed shuffling. We also used ARIADNE [12], that uses an approximate formula to estimate gapped K and λ from ungapped K and λ. Both these methods are currently applicable only for alignment methods using one parameter set. We also used the linear regression fitting program used in [9] to estimate K and λ from an empirical distribution of alignment scores. Finally, we also used the maximum likelihood method [23] and corresponding routines in the HMMER package [22] to fit an EVD to the empirical distribution. We compared all these methods on the basis of accuracy in estimating K and λ values for a pair of sequences.

4 Experiments and Results

4.1 Accurate Estimation of K and for λ a Specific Sequence Pair

For each sequence pair, we need to find accurate estimates of the statistical parameters K and λ. Here, we are not too much concerned with the time taken for estimating K and λ since we are interested in determining the method which gives the most accurate estimates of the parameters. Therefore, we can afford to spend more time for accurate estimates.

To decide on the method for estimating statistical parameters for a sequence pair, we used the following approach: a pair of remotely homologous protein sequences was selected using PSI-BLAST by giving a G protein-coupled receptor sequence (GENE ID: 55507 GPRC5D) as query and running two iterations of PSI-BLAST. The second sequence was selected from the new results after second iterations that were not present in the results of the first iteration. The sequence was a novel protein similar to vertebrate pheromone receptor protein [Danio rerio] (emb|CAM56437.1|). We used this pair of real protein sequences to generate eleven large scale simulations of alignment score distributions using

different alignment programs and scoring schemes described in Section 3. Each of the eleven simulations involved aligning one million pairs of randomly shuffled versions of the sequence pair (with different seeds for the random number generator). Because we are mostly interested in the tail distribution of scores, we looked at the distribution of scores for which the normalized E-value was less than 0.01. We got eleven empirically derived random distributions, and although theoretically they should have been same, there was slight variation within the eleven distributions (because of random sampling). Here we combined the eleven distributions by taking the mean of the E-values for each score from each of the eleven distributions. This is equivalent to doing one big simulation with eleven million shuffles. We assume that the resulting mean distribution is the most accurate representation of the actual distribution and subsequently used this distribution to validate the predicted E-values from different methods of estimating K and λ. Fig. 1 shows the mean score distribution (complementary distribution function in terms of statistics) based on the simulations, which is same as the normalized E-value, for three alignment schemes. The solid line curve shows the mean of the normalized E-values from the eleven different simulations. The vertical bars for each alignment score indicates the variation in normalized E-values observed within the eleven different simulations.

For evaluating various methods of estimating statistical parameters, the K and λ estimates from different programs for the same sequence pair were examined. For the PRSS program, both uniform and windowed shuffling was used with two values of window size: 10 and 20. The ARIADNE program was also used to estimate gapped K and λ. Since we are interested in accurate fitting of the tail distribution, for the curve fitting methods like maximum likelihood (ML) and linear regression (LR), we used the censored distribution for fitting. Here type-I censoring is defined as the one in which we fit only the data right of the peak of the histogram [23], and type-II censoring is defined as one where the cutoff is set to the score that corresponds to a normalized E-value of 0.01. We also show results for uncensored fitting with ML method, applied to the eleven empirical distributions (with a million shuffles each) to make a realistic comparison of other fitting schemes with the methodology used in PRSS, which also uses maximum likelihood method, but only up to 1000 shuffles. Since we generated eleven independent score distributions, we used them individually to estimate eleven pairs of K and λ using both ML and LR, so that we can perform the best case, worst case and average case prediction analysis for fitting methods. The estimated K and λ values from each program are used to predict the E-values for different alignment scores using the EVD formula, and the resulting distribution is compared with the mean empirical distribution generated from eleven independent simulations as described above.

Table 1 shows the comparison of the sum of squares of differences (SSD) between predicted normalized E-values and actual normalized E-values for different methods and alignment schemes. Since we had eleven estimates of K and λ for the ML and LR methods, we report the minimum, maximum and average SSD. PRSS and ARIADNE report one set of parameters, and thus there is

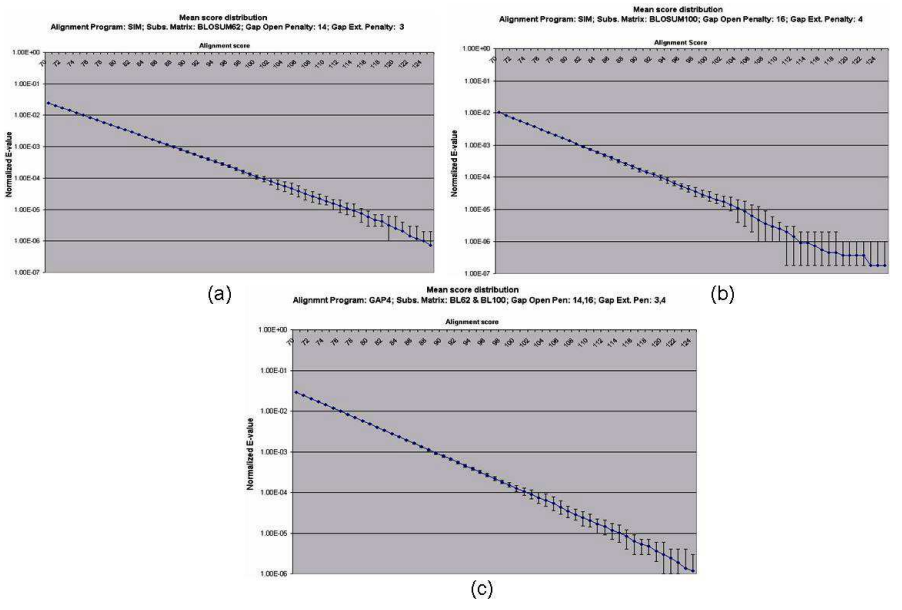


Fig. 1. Distribution of alignment scores generated (a) using SIM program and BLOSUM62 matrix, (b) using SIM program and BLOSUM100 matrix and (c) using GAP4 program and BLOSUM62 and BLOSUM100 matrices. The solid line curve represents the mean of the eleven distributions generated, and the vertical bars represent the variation within the eleven distributions.

Table 1. Comparison of the Sum of Squares of Differences (SSD) between predicted normalized E-values and actual normalized E-values for different methods and alignment schemes

Program: SIM		Matrix: BLOSUM62			GapOpenPen.: 14, GapExtPen.: 3				
Statistic	Ariadne	PRSS			Maximum Likelihood			LinRegr	Minimum
		Uniform	-w 10	-w 20	Full	Censor-I	Censor-II	Censor-II	
Min(SSD)					**8.05E-09**	9.11E-09	2.67E-08	8.58E-08	**8.05E-09**
Max(SSD)	5.6×	3.46×	4.22×	7.5×	6.03E-07	**2.75E-07**	2.15E-06	5.20E-06	**2.75E-07**
Avg(SSD)	E-04	E-05	E-02	E-03	3.02E-07	**7.91E-08**	6.08E-07	1.48E-06	**7.91E-08**
Program: SIM		Matrix: BLOSUM100			GapOpenPen.: 16, GapExtPen.: 4				
Statistic	Ariadne	PRSS			Maximum Likelihood			LinRegr	Minimum
		Uniform	-w 10	-w 20	Full	Censor-I	Censor-II	Censor-II	
Min(SSD)					1.88E-09	1.76E-09	**8.16E-10**	8.27E-09	**8.16E-10**
Max(SSD)	1.02×	4.58×	8.3×	4.38×	3.90E-08	**2.50E-08**	1.62E-07	4.20E-07	**2.50E-08**
Avg(SSD)	E-05	E-05	E-04	E-04	**8.51E-09**	9.18E-09	4.54E-08	1.13E-07	**8.51E-09**
Program: GAP4		Matrix: BL62,BL100			GapOpen:14,16 GapExt:3,4				
Statistic	Ariadne	PRSS			Maximum Likelihood			LinRegr	Minimum
		Uniform	-w 10	-w 20	Full	Censor-I	Censor-II	Censor-II	
Min(SSD)					2.20E-07	2.05E-08	**1.35E-08**	9.34E-08	**1.35E-08**
Max(SSD)	NA	NA	NA	NA	1.62E-06	**6.86E-07**	2.97E-06	9.77E-06	**6.86E-07**
Avg(SSD)					9.88E-07	**2.42E-07**	6.49E-07	2.83E-06	**2.42E-07**

only one SSD corresponding to these methods. Further, for alignment method GAP4 which can use multiple parameter sets, there is no entry corresponding to ARIADNE and PRSS, as these methods do not currently support the use of multiple parameter sets. The last column gives the minimum SSD obtained, and its second and third entries correspond to the minimum worst case and minimum average case error in prediction. We can see that the minimum SSD is obtained for the ML method in all cases. Specifically, ML fitting with type-I censoring gives the minimum Max(SSD), (i.e. minimum worst case error) for all the three cases. Therefore, we conclude that ML fitting with type-I censoring gives the most accurate estimates of statistical parameters K and λ.

4.2 Using Pairwise Statistical Significance to Infer Homology

To evaluate our method, we used a non-redundant subset of the CATH 2.3 database (Class, Architecture, Topology, and Hierarchy, [29]) provided by [24] and available at ftp://ftp.ebi.ac.uk/pub/software/unix/fasta/prot_sci_04/. As described in [24], this dataset consists of 2771 domain sequences and includes 86 selected test query sequences, each representing at least five members of their respective CATH sequence family (35% sequence identity) in the data set. We used this database and query set for experimenting with pairwise statistical significance. For each of the 86×2771 comparisons, we used the maximum likelihood method with type-1 censoring with 2000 shuffles to fit the score distribution from the GAP3 program with a very high difference block penalty (to not use that feature), which essentially reduces it to an ordinary alignment program like SIM. Alignments were obtained using the BLOSUM50 substitution matrix (in 1/3 bit units as used by SSEARCH) with gap open penalty as 10, and gap extension penalty as 2. The same combination of parameters was used in [24] to report the results obtained with the SSEARCH program. The parameters K and λ resulting from the ML fitting were then used to find the pairwise statistical significance of the pairwise comparison, and the P-value was recorded. Following [24], Error per Query (EPQ) versus Coverage plots were used to present the results. To create these plots, the list of pairwise comparisons were sorted based on statistical significance, and subsequently, the lists were examined, from best score to worst. Going down the list, the coverage count is increased by one if the two members of the pair are homologs, and the error count is increased by one if they are not. At a given point in the list, EPQ is the total number of errors incurred so far, divided by the number of queries. Coverage at that point is the fraction of homolog pairs detected at this significance level.

For each of the 86 queries, 2771 comparisons were done, and EPQ vs. Coverage curves were plotted. Since the EPQ vs. Coverage curves on the complete dataset can be distorted due to poor performance by one or two queries (if those queries produce many errors at low coverage levels), reference [24] examined the performance of the methods with individual queries. Fig. 2(a) shows the level of coverage generated by the median query (43 queries performed better, 43 worse) at the 1st, 3rd, 10th, 30th, and 100th false positive for homologs. Fig. 2(b) shows the same results for 25th percentile of coverage (i.e. 21 of the queries have worse

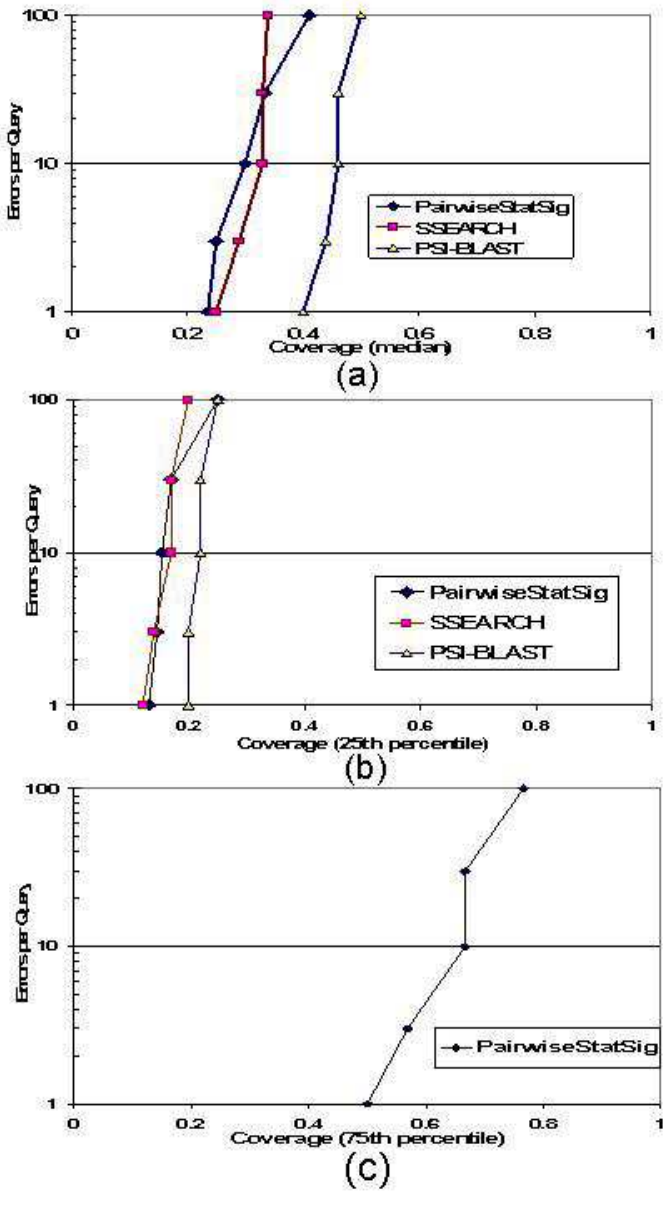


Fig. 2. Errors per Query vs. Coverage plots for individual queries. (a) The median level of coverage for 86 queries; (b) 1st quartile (25th percentile) coverage; (c) 3rd quartile (75th percentile) coverage. Along with the curve for pairwise statistical significance, the curves for SSEARCH and PSI-BLAST in (a) and (b) are derived from figures 2A and 2B in [24]. The corresponding results for (c) were not available in [24].

coverage, and 65 have better coverage). And fig. 2(c) shows the same results for 75th percentile of coverage (i.e. 65 of the queries have worse coverage, and 21 have better coverage). Along with the curve for pairwise statistical significance, the curves for SSEARCH and PSI-BLAST in fig. 2(a) and (b) are derived from the figures 2A and 2B in [24]. The results corresponding to Fig. 2(c) were not available in [24], and hence, only the results of pairwise statistical significance are reported. This figure shows that pairwise statistical significance performs comparable to and sometimes significantly better than database statistical significance (with SSEARCH program), particularly at higher error rates. However, the results using PSI-BLAST are clearly the best.

Since the SSEARCH program used the same substitution matrix as we used for our experiments (BLOSUM50) [24], the results indicate that pairwise statistical significance works better in practice than database statistical significance. However, even better results with PSI-BLAST using database statistical significance indicates that sequence specific substitution matrices should be used for the pairwise comparisons, and to fairly compare pairwise statistical significance with the database statistical significance reported by PSI-BLAST, more experiments need to be performed with pairwise statistical significance using sequence specific substitution matrices.

The time required to estimate pairwise statistical significance for a given pair of sequences is certainly expected to depend on the length of the two sequences. Therefore, to get an idea of the average time needed to estimate pairwise statistical significance using the proposed method, we used the following approach. We took a real sequence from the CATH2.3 database of length 135 (1que01) and estimated its pairwise statistical significance with more than a thousand other real sequences. It took 2574.151 seconds for finding 1013 pairwise statistical significance estimates on an Intel processor 2.8GHz, which means on an average 2.54 seconds per comparison. Certainly, this is much faster than a database search, if we are only interested in a specific (or a few) pairwise comparison(s), but will take a huge amount of time if applied for all pairwise comparisons in a large database search.

The program PairwiseStatSig is available for free academic use at www.cs.iastate.edu/∼ankitag/PairwiseStatSig.html

5 Conclusion and Future Work

This paper explores the use of pairwise statistical significance, and compares it with database statistical significance for the application of homology detection. Large scale experimentation was done to determine the most accurate method for determining pairwise statistical significance. Further, preliminary experimentation for homology detection with a benchmark database (a subset of CATH2.3 database) shows that the pairwise statistical significance performs better than database statistical significance (using SSEARCH program), but still the accuracy of retrieval results is best for PSI-BLAST.

We believe that PSI-BLAST gives best results because of the use of sequence specific substitution matrices, although it also uses database statistical significance to estimate the E-value. Using pairwise statistical significance is shown to be better than database E-value (used in SSEARCH), and thus, we believe that the results of pairwise statistical significance can be further improved by using sequence specific substitution matrices, which is the significant part of our future work. Also, more experimentation with other standard databases such as SCOP can be done to compare the performance. Another major contribution can be to estimate the pairwise statistical significance accurately in less time, as the method used in this paper was to use maximum likelihood to fit a score distribution generated by simulation, which is not time-efficient. Faster methods for determining pairwise statistical significance are thus required. We have made some progress in this direction [30]. Another aspect of future work is to experiment with other sample space for shuffling of protein sequences for generating score distribution, which may provide better significance estimates.

References

1. Pearson, W.R., Lipman, D.J.: Improved Tools for Biological Sequence Comparison. Proceedings of the National Academy of Sciences, USA 85(8), 2444–2448 (1988)
2. Altschul, S.F., Gish, W., Miller, W., Myers, E.W., Lipman, D.J.: Basic Local Alignment Search Tool. Journal of Molecular Biology 215(3), 403–410 (1990)
3. Altschul, S.F., Madden, T.L., Schäffer, A.A., Zhang, J., Zhang, Z., Miller, W., Lipman, D.J.: Gapped BLAST and PSI-BLAST: A New Generation of Protein Database Search Programs. Nucleic Acids Research 25(17), 3389–3402 (1997)
4. Smith, T.F., Waterman, M.S.: Identification of Common Molecular Subsequences. Journal of Molecular Biology 147(1), 195–197 (1981)
5. Sellers, P.H.: Pattern Recognition in Genetic Sequences by Mismatch Density.. Bulletin of Mathematical Biology 46(4), 501–514 (1984)
6. Pearson, W.R.: Effective Protein Sequence Comparison. Methods in Enzymology 266, 227–259 (1996)
7. Pearson, W.R.: Flexible Sequence Similarity Searching with the FASTA3 Program Package.. Methods in Molecular Biology 132, 185–219 (2000)
8. Huang, X., Chao, K.-M.: A Generalized Global Alignment Algorithm. Bioinformatics 19(2), 228–233 (2003)
9. Huang, X., Brutlag, D.L.: Dynamic Use of Multiple Parameter Sets in Sequence Alignment. Nucleic Acids Research 35(2), 678–686 (2007)
10. Karlin, S., Altschul, S.F.: Methods for Assessing the Statistical Significance of Molecular Sequence Features by Using General Scoring Schemes. Proceedings of the National Academy of Sciences, USA 87(6), 2264–2268 (1990)
11. Mott, R., Tribe, R.: Approximate Statistics of Gapped Alignments. Journal of Computational Biology 6(1), 91–112 (1999)
12. Mott, R.: Accurate Formula for P-values of Gapped Local Sequence and Profile Alignments. Journal of Molecular Biology 300, 649–659 (2000)
13. Altschul, S.F., Bundschuh, R., Olsen, R., Hwa, T.: The estimation of statistical parameters for local alignment score distributions. Nucleic Acids Research 29(2), 351–361 (2001)

14. Schäffer, A.A., Aravind, L., Madden, T.L., Shavirin, S., Spouge, J.L., Wolf, Y.I., Koonin, E.V., Altschul, S.F.: Improving the Accuracy of PSI-BLAST Protein Database Searches with Composition-based Statistics and Other Refinements. Nucleic Acids Research 29(14), 2994–3005 (2001)
15. Yu, Y.K., Gertz, E.M., Agarwala, R., Schäffer, A.A., Altschul, S.F.: Retrieval Accuracy, Statistical Significance and Compositional Similarity in Protein Sequence Database Searches. Nucleic Acids Research 34(20), 5966–5973 (2006)
16. Kschischo, M., Lässig, M., Yu, Y.: Toward an Accurate Statistics of Gapped Alignments. Bulletin of Mathematical Biology 67, 169–191 (2004)
17. Grossmann, S., Yakir, B.: Large Deviations for Global Maxima of Independent Superadditive Processes with Negative Drift and an Application to Optimal Sequence Alignments. Bernoulli 10(5), 829–845 (2004)
18. Pearson, W.R., Wood, T.C.: Statistical Significance in Biological Sequence Comparison. In: Balding, D.J., Bishop, M., Cannings, C. (eds.) Handbook of Statistical Genetics, pp. 39–66. Wiley, Chichester, UK (2001)
19. Mott, R.: Alignment: Statistical Significance. Encyclopedia of Life Sciences (2005), http://mrw.interscience.wiley.com/emrw/9780470015902/els/article/a0005264/current/abstract
20. Mitrophanov, A.Y., Borodovsky, M.: Statistical Significance in Biological Sequence Analysis. Briefings in Bioinformatics 7(1), 2–24 (2006)
21. Pearson, W.R.: Empirical Statistical Estimates for Sequence Similarity Searches. Journal of Molecular Biology 276, 71–84 (1998)
22. Eddy, S.R.: Multiple Alignment Using Hidden Markov Models. In: Rawlings, C., Clark, D., Altman, R., Hunter, L., Lengauer, T., Wodak, S. (eds.) Proceedings of the Third International Conference on Intelligent Systems for Molecular Biology, pp. 114–120. AAAI Press, Menlo Park (1995)
23. Eddy, S.R.: Maximum Likelihood Fitting of Extreme Value Distributions (1997), unpublished manuscript, citeseer.ist.psu.edu/370503.html
24. Sierk, M.L., Pearson, W.R.: Sensitivity and Selectivity in Protein Structure Comparison. Protein Science 13(3), 773–785 (2004)
25. Wolfsheimer, S., Burghardt, B., Hartmann, A.K.: Local Sequence Alignments Statistics: Deviations from Gumbel Statistics in the Rare-event Tail. Algorithms for Molecular Biology 2(9) (2007), http://www.almob.org/content/2/1/9
26. Altschul, S.F., Gish, W.: Local Alignment Statistics. Methods in Enzymology 266, 460–480 (1996)
27. Olsen, R., Bundschuh, R., Hwa, T.: Rapid Assessment of Extremal Statistics for Gapped Local Alignment. In: Proceedings of the Seventh International Conference on Intelligent Systems for Molecular Biology, pp. 211–222. AAAI Press, Menlo Park (1999)
28. Huang, X., Miller, W.: A Time-efficient Linear-space Local Similarity Algorithm. Advances in Applied Mathematics 12(3), 337–357 (1991)
29. Orengo, C.A., Michie, A.D., Jones, S., Jones, D.T., Swindells, M.B., Thornton, J.M.: CATH - A Hierarchic Classification of Protein Domain Structures. Structure 28(1), 1093–1108 (1997)
30. Agrawal, A., Ghosh, A., Huang, X.: Estimating Pairwise Statistical Significance of Protein Local Alignments Using a Clustering-Classification Approach Based on Amino Acid Composition. In: Măndoiu, I., Sunderraman, R., Zelikovsky, A. (eds.) ISBRA 2008. LNCS(LNBI), vol. 4983, pp. 62–73. Springer, Heidelberg (2008)

Estimating Pairwise Statistical Significance of Protein Local Alignments Using a Clustering-Classification Approach Based on Amino Acid Composition

Ankit Agrawal[1], Arka Ghosh[2], and Xiaoqiu Huang[1]

[1] Department of Computer Science, Iowa State University,
226 Atanasoff Hall, Ames, IA 50011-1041, USA
{ankitag,xqhuang}@iastate.edu
[2] Department of Statistics, Iowa State University, 303 Snedecor Hall
Ames, IA, 50011-1210, USA
apghosh@iastate.edu

Abstract. A central question in pairwise sequence comparison is assessing the statistical significance of the alignment. The alignment score distribution is known to follow an extreme value distribution with analytically calculable parameters K and λ for ungapped alignments with one substitution matrix. But no statistical theory is currently available for the gapped case and for alignments using multiple scoring matrices, although their score distribution is known to closely follow extreme value distribution and the corresponding parameters can be estimated by simulation. Ideal estimation would require simulation for each sequence pair, which is impractical. In this paper, we present a simple clustering-classification approach based on amino acid composition to estimate K and λ for a given sequence pair and scoring scheme, including using multiple parameter sets. The resulting set of K and λ for different cluster pairs has large variability even for the same scoring scheme, underscoring the heavy dependence of K and λ on the amino acid composition. The proposed approach in this paper is an attempt to separate the influence of amino acid composition in estimation of statistical significance of pairwise protein alignments. Experiments and analysis of other approaches to estimate statistical parameters also indicate that the methods used in this work estimate the statistical significance with good accuracy.

Keywords: Clustering, Classification, Pairwise local alignment, Statistical significance.

1 Introduction

Sequence alignment is extremely useful in the analysis of DNA and protein sequences [1]. Sequence alignment forms the basic step of making various high level inferences about the DNA and protein sequences - like homology, finding protein function, protein structure, deciphering evolutionary relationships, etc.

I. Măndoiu, R. Sunderraman, and A. Zelikovsky (Eds.): ISBRA 2008, LNBI 4983, pp. 62–73, 2008.

There are many programs that use some well known algorithms [2,3] or their heuristic version [1,4,5]. Recently, some enhancements in alignment program features have also become available [6,7] using difference blocks and multiple scoring matrices. Quality of a pairwise sequence alignment is gauged by the statistical significance rather than the alignment score alone, i.e., if an alignment score has a low probability of occurring by chance, the alignment is considered statistically significant.

For ungapped alignments, rigorous statistical theory for the alignment score distribution is available [8], and it was shown that the statistical parameters K and λ can be calculated analytically for a pair of sequences with given amino acid composition and scoring scheme. However, no perfect theory currently exists for gapped alignment score distribution, and for score distributions from alignment programs using additional features like difference blocks [6], and which use multiple parameter sets [7]. The problem of accurately determining the statistical significance of gapped sequence alignment has attracted a lot of attention in the recent years [9,10,11,12,13,14,15]. There exist a couple of good starting points for statistically describing gapped alignment score distributions [16,17], but a complete mathematical description of the optimal scores distribution remains far from reach [17]. Some excellent reviews on statistical significance in sequence comparison are available in the literature [18,19,20].

The statistical significance of a pairwise alignment depends upon various factors sequence alignment method, scoring scheme, sequence length, and sequence composition [19]. The straightforward way to estimate statistical significance of scores from an alignment program for which the statistical theory is unavailable is to generate a distribution of alignment scores using the program with randomly shuffled versions of the pair of sequences, and compare the obtained score with the generated score distribution, either directly or by fitting an extreme value distribution (EVD) curve (explained in the next section) to the generated distribution to get the EVD parameters K and λ, and using the EVD formula with the estimated K and λ to calculate the statistical significance of the obtained score. However, the parameters thus obtained are ideally valid only for the specific sequence pair under consideration, and for any other sequence pair, the parameters should be recomputed by generating another distribution, which is very time-consuming and impractical.

Thus, BLAST2.0 [1] uses a lookup method wherein the parameters K and λ are pre-computed for different scoring schemes assuming average amino acid composition of both sequences. PRSS program in the FASTA package [4,5,9] calculates the statistical significance of an alignment by aligning them, shuffling the second sequence up to 1000 times, and estimating the statistical significance from the distribution of shuffled alignment scores. It uses maximum likelihood to fit an EVD to the shuffled score distribution. A similar approach is also used in HMMER [21]. It also uses maximum likelihood fitting [22] and also allows for censoring of data left of a given cutoff, for fitting only the right tail of the histogram. A heuristic approximation of the gapped local alignment score distribution is also available [10], and based on these statistics, accurate formulae

for statistical parameters K and λ for gapped alignments are derived and implemented in a program called ARIADNE [11]. These methods can provide an accurate estimation of statistical significance for gapped alignments, but currently do not incorporate the additional features of sequence alignment, like using difference blocks and multiple parameter sets [6,7].

The problem of estimating the statistical significance of the database searches has been addressed in much detail over the past two decades as discussed earlier. However, accurate estimation of the statistical significance of specific pairwise alignments needs directed research efforts. It is an important problem critical in comparison of various alignment programs, and especially with new alignment programs coming up with additional features to suit the features of the real biological sequences, this problem of estimating statistical significance for pairwise sequence alignments becomes particularly important. It has also been shown recently [23] that pairwise statistical significance is a better indicator of homology than database statistical significance. The method used in [23], although was shown to be accurate, but is also very time-consuming, as it involves generating a score distribution of tens of thousands of alignments. The need for faster methods for estimating pairwise statistical significance was also stressed in [23].

In this paper, we propose and implement a simple clustering-classification approach that clusters the universe of protein sequences based on amino acid composition, and estimates the parameters K and λ for all cluster pairs for different scoring schemes and alignment methods. In this way, we attempt to separate the dependence of the statistical parameters K and λ on amino-acid composition from other factors like alignment method and scoring schemes. The task of estimating statistical significance thus reduces to classifying the sequences to appropriate clusters, and using the corresponding K and λ values of the classified cluster pair. This approach is similar to the lookup method used in BLAST2.0 [1] but takes into account the features of the specific sequence pair being aligned. For simple alignment methods, the results are also presented using other approaches (PRSS [4,5,9] and ARIADNE [11]), and for advanced alignment methods [6,7] currently no other quick methods are available to estimate pairwise statistical significance except the method described in this paper.

2 The Extreme Value Distribution for Ungapped and Gapped Alignments

Just as the distribution of the sum of a large number of independent identically distributed (i.i.d) random variables tends to a normal distribution (central limit theorem), the distribution of the maximum of a large number of i.i.d. random variables tends to an extreme value distribution (EVD). This is an important fact, because it allows us to fit an EVD to the score distribution from any local alignment program, and use it for estimating statistical significance of scores from that program. The distribution of Smith-Waterman local alignment score between random, unrelated sequences is approximately a Gumbel-type EVD [8]. In the limit of sufficiently large sequence lengths m and n, the statistics of HSP

(High-scoring Segment Pairs which correspond to the ungapped local alignment) scores are characterized by two parameters, K and λ. The probability that the optimal local alignment score S exceeds x is given by

$$\Pr(S > x) \sim 1 - exp[-Kmne^{-\lambda x}]$$

This is valid for ungapped alignments [8], and the parameters K and λ can be computed analytically from the substitution scores and sequence compositions. An important point here is that this scheme allows for the use of only one substitution matrix. For the gapped alignment, no perfect statistical theory has yet been developed, although there is ample evidence that it also closely follows an extreme value distribution [9,11,24,7].

3 Clustering-Classification Approach

This paper presents a simple clustering-classification approach based on amino acid composition for estimating statistical significance of pairwise protein local alignments, which is essentially an enhanced lookup method, where K and λ values are pre-computed for each cluster pair by simulation. Subsequently, for a given sequence pair, the sequences are individually classified to the corresponding clusters based on their amino acid composition, and the K and λ parameters for the cluster pair are used for statistical significance calculation of alignments of the sequence pair.

3.1 Clustering

There are many algorithms available for clustering like hierarchical clustering, k-means clustering, etc. [25]. Here we are dealing with clustering the universe of protein sequences whose number is in hundreds of thousands. Therefore, we use k-means clustering as hierarchical methods typically involve the computation of a distance matrix of quadratic complexity with respect to the input size. In this work, we have used the k-means implementation in R package [26]. Each of the k clusters of sequences is represented by a single representative sequence (central sequence), and subsequently the parameters K and λ are computed for each pair of the k representative sequences. Given below a pseudo code for the clustering module:

```
alphabet = "ACDEFGHIKLMNPQRSTVWY"   #protein alphabet (amino acids)
sequences4R = set of sequences to be clustered
nSeq = number of sequences
for (i in 1:nSeq) {
  seqArray = sequences4R[i]
  lenSeq=length(seqArray)
  for (j in 1:lenAlphabet-1) {
    AACounts[i,j] = number of occurrences of amino acid alphabet[j] in seqArray
  }
  AAComposition[i,]=AACounts[i,]/lenSeq
}
```

```
k = number of clusters
seqClusters = clustered sequences based on AAComposition
for (i in 1:k) {
  clust_reprSeq[i] = representative sequence of cluster[i]
}
for (i in 1:k) {
  for (j in 1:i) {
    Compute the value of K and lambda by empirical simulation
    K_clusters[i,j] = K_clusters[j,i] = estimated K
    lambda_clusters[i,j] = lambda_clusters[j,i] = estimated lambda
  }
}
```

3.2 Classification

Given two protein sequences for estimation of statistical parameters, they are classified individually to the appropriate clusters. Each of the k clusters obtained in the clustering step have their center, which corresponds to the central amino acid composition for that cluster. A sequence is classified to the cluster that minimizes the sum of squares of differences between the amino acid composition of the sequence and the central amino acid composition of the cluster. Subsequently, the pre-computed K and λ values for the classified cluster pair are used for the statistical significance estimation of alignments of the two input sequences. Given below a pseudo code for the classification module:

```
alphabet = "ACDEFGHIKLMNPQRSTVWY"
sequences4R = set of two sequences for which K and lambda is to be estimated
nSeq = 2
for (i in 1:nSeq) {
  seqArray = sequences4R[i]
  lenSeq=length(seqArray)
  for (j in 1:lenAlphabet-1) {
    AACounts[i,j] = number of occurrences of amino acid alphabet[j] in seqArray
  }
  AAComposition[i,]=AACounts[i,]/lenSeq
}
k = number of clusters
for (j in 1:nSeq) {
  classifiedCluster[j] = classified cluster based on AAComposition
}
estimatedK = K_clusters[classifiedCluster[1],classifiedCluster[2]]
estimatedLambda=lambda_clusters[classifiedCluster[1],classifiedCluster[2]]
```

4 Tools and Programs Used

We worked with the alignment programs SIM [27], which is an ordinary alignment program (similar to SSEARCH), and GAP4 [7], which allows dynamically

finding similarity blocks and difference blocks [6], as well as using multiple parameter sets (scoring matrices, gap penalties, difference block penalties) to generate a single pairwise alignment. For estimating the statistical parameters K and λ, we used several programs. First is PRSS from the FASTA package [4,5,9], which takes two protein sequences and one set of parameters (scoring matrix, gap penalty), generates the optimal alignment, and estimates the K and λ parameters by aligning up to 1000 shuffled versions of the second sequence, and fitting an EVD using Maximum Likelihood. In addition to uniform shuffling, it also allows for windowed shuffling. We also used ARIADNE [11], that uses an approximate formula to estimate gapped K and λ from ungapped K and λ, which are calculable analytically as described before. Both these methods are currently applicable only for alignment methods using one parameter set. We also used the Linear Regression fitting program used in [7] to estimate K and λ from an empirical distribution of alignment scores. Finally, we also used the Maximum likelihood method [22] and corresponding routines in the HMMER package [21] to fit an EVD to the empirical distribution. Here type-I censoring is defined as the one in which we fit only the data right of the peak of the histogram [22], and type-II censoring is defined as one where the cutoff is set to the score that corresponds to a normalized E-value of 0.01. We used all these methods to estimate K and λ values for a pair of representative sequences for a given alignment scheme.

5 Experiments and Results

We downloaded all the available 261513 Swissprot protein sequences from http://www.ebi.ac.uk/FTP/. The statistics of the lengths of the sequences are given in Table 1, and the histogram of sequence lengths less than 1000 is shown in Fig. 1. Clearly, the variation in sequence length is very extreme, although the length of most of the sequences is in the range of 150 to 450. To minimize the influence of variation in length, we only select the sequences with length between the 1st and 3rd quartile for clustering. The number of sequences between 1st and 3rd quartile is 131486. The amino acid composition of all these sequences is calculated, and an implementation of the k-means clustering algorithm in R package [26] is used to cluster the sequences into k=5 clusters, based on their amino acid composition. The k-means implementation in R returns for each of the k clusters its center, its within-sum-of-squares, its size, and of course, the classification of the input data points in one of the k clusters.

Fig. 2 is an attempt to visualize the clusters by representing the 20 dimensional amino-acid-composition vector as a point in $x-y$ plane using the first two amino-acid-compositions. Although it does not give a full picture of the clusters and their separation, it nonetheless gives some idea of how the clusters are located. One representative sequence for each of the k clusters is then selected by choosing the one whose amino-acid-composition vector is the closest to the center of the cluster (i.e., which gave the minimum sum of square of differences). Then, for each pair of the representative sequences, the parameters K and λ are

estimated using the methods described earlier. This work presents the preliminary analysis taking k as 5 to study the effectiveness of this method. However, no detailed study on the number of clusters has been presented in this work. For k

Table 1. Statistics of lengths of sequences

Minimum	1st Quartile	Median	Mean	3rd Quartile	Maximum
2	165	296	365.7	460	34350

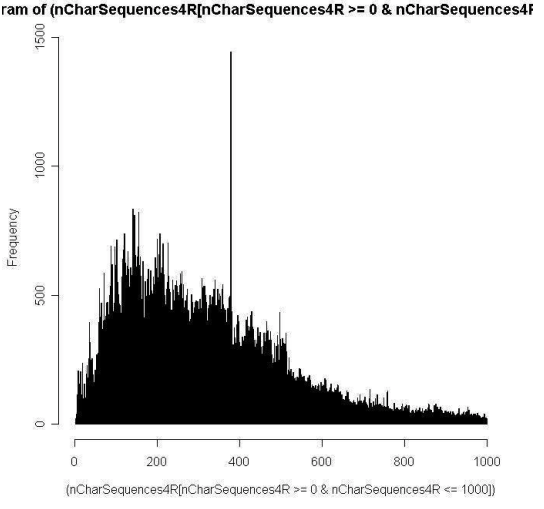


Fig. 1. Histogram of length of sequences with length ≤ 1000

$= 5$, there exist 15 ($=^5 C_2 + 5$) different pairwise cluster combinations. Table 2 gives the K and λ estimates for one of the 15 pairwise clusters ($\langle 3, 2 \rangle$). Here, we used several options for the alignment parameters. For substitution matrices, we used all possible combinations of BLOSUM45, BLOSUM62, and BLOSUM100 matrices. The alignment program GAP4 [7] is capable of using multiple substitution matrices to produce a single optimal alignment of two sequences. It requires all substitution matrices to be in the same scale, and thus all matrices were used in 1/3 bit scale. Other parameters like gap penalties, etc. were the same as the default values used in GAP4 [7] for matrices in 1/3 bit scale. We used the various programs for statistical parameter estimation as described earlier. Rows in first half of Table 2 show the K and λ estimates from ARIADNE [11] and PRSS [4,5,9], and the second half of the table show the estimates from ML and LR. As pointed out earlier, ARIADNE and PRSS currently can work only with one parameter set, and cannot estimate the pairwise statistical significance parameters for alignment programs that use multiple parameter sets, and hence, the corresponding entries in Table 2 are not available.

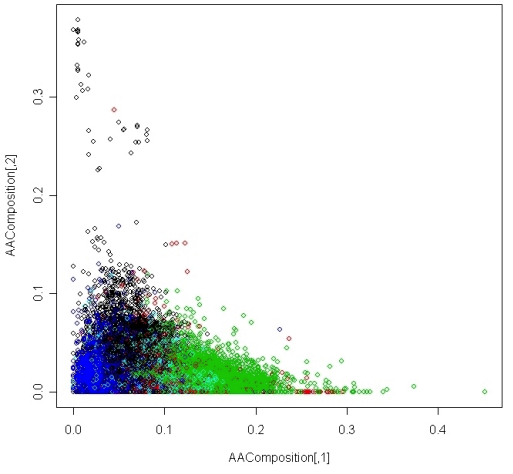


Fig. 2. k-means clusters (k=5). Sequences in each cluster are represented by different colors. This visualization represents the 20-dimensional amino-acid-composition vector by a 2-dimensional vector (corresponding to the first two entries of the 20-dimensional amino-acid-composition vector), and hence is not complete, but gives an overall idea of how the clusters are located.

Table 2. K and λ estimates for the cluster pair $\langle 3, 2 \rangle$

Substitution Matrix	Gap Open	Gap Ext	ARIADNE		PRSS(1000 shuffles)					
			K	λ	uniform		-w 10		-w 20	
					K	λ	K	λ	K	λ
BLOSUM45	12	2	0.01795	0.184148	0.0329	0.1869	0.03736	0.1941	0.0381	0.1974
BLOSUM62	14	3	0.06445	0.200311	0.0956	0.2104	0.1108	0.2154	0.1212	0.2181
BLOSUM100	16	4	0.15101	0.210326	0.1888	0.224	0.2624	0.2328	0.1564	0.2198
BL45,62,100	12,14,16	2,3,4	NA	NA	NA	NA	NA	NA	NA	NA
BL45,BL62	12,14	2,3	NA	NA	NA	NA	NA	NA	NA	NA
BL45,BL100	12,16	2,4	NA	NA	NA	NA	NA	NA	NA	NA
BL62,BL100	14,16	3,4	NA	NA	NA	NA	NA	NA	NA	NA

Substitution Matrix	Gap Open	Gap Ext	Maximum Likelihood (100000 shuffles)						LinearRegr. (100000 shfls)	
			Full		Censored-I		Censored-II			
			K	λ	K	λ	K	λ	K	λ
BLOSUM45	12	2	0.03387	0.189248	0.0316	0.1876	0.089487	0.204045	0.1083	0.2063
BLOSUM62	14	3	0.08757	0.205953	0.0875	0.2058	0.045709	0.196304	0.2389	0.2195
BLOSUM100	16	4	0.18503	0.2191	0.1761	0.2179	0.358664	0.228915	0.4009	0.2304
BL45,62,100	12,14,16	2,3,4	0.10576	0.194163	0.0967	0.1923	0.096223	0.192396	0.2358	0.2044
BL45,BL62	12,14	2,3	0.06773	0.194176	0.0919	0.1982	0.123769	0.202883	0.1551	0.2057
BL45,BL100	12,16	2,4	0.09969	0.195183	0.0911	0.1932	0.147417	0.200051	0.3205	0.2102
BL62,BL100	14,16	3,4	0.15685	0.207436	0.1570	0.2074	0.243203	0.21407	0.2807	0.2157

It was reported in [23] that that Maximum likelihood fitting with type-I censoring gives the most accurate estimates of K and λ for estimation of pairwise statistical significance. Therefore, we report the corresponding the K and λ estimates for all cluster-pairs in Table 3, presenting the final result of this work. There are 7 sub-tables in Table 3, each showing the K and λ estimates for all cluster pairs for a unique scoring scheme (7 scoring schemes are presented here).

Table 3. Pairwise cluster statistical significance parameters for a variety of scoring schemes

Parameters	Substitution Matrix: BLOSUM45; Gap Open Penalty: 12; Gap Extension Penalty: 2									
	λ					K				
Cluster	1	2	3	4	5	1	2	3	4	5
1	0.1358571					0.036457				
2	0.212847	0.1642628				0.055198	0.022692			
3	0.2076972	0.187666	0.1501918			0.052404	0.031631	0.020645		
4	0.2439074	0.1868858	0.2037552	0.1584317		0.070919	0.031597	0.040751	0.018778	
5	0.1948708	0.1909384	0.190503	0.1971262	0.1733189	0.041417	0.033241	0.032547	0.034713	0.024396

Parameters	Substitution Matrix: BLOSUM62; Gap Open Penalty: 14; Gap Extension Penalty: 3									
	λ					K				
Cluster	1	2	3	4	5	1	2	3	4	5
1	0.155231					0.053164				
2	0.2214101	0.1904642				0.108926	0.076897			
3	0.2173865	0.2058921	0.1772875			0.11316	0.087505	0.059218		
4	0.2461108	0.209065	0.2199229	0.1976537		0.12987	0.091884	0.104014	0.085541	
5	0.2085433	0.2057725	0.2063069	0.2142532	0.193569	0.09491	0.087126	0.089158	0.095499	0.069993

Parameters	Substitution Matrix: BLOSUM100; Gap Open Penalty: 16; Gap Extension Penalty: 4									
	λ					K				
Cluster	1	2	3	4	5	1	2	3	4	5
1	0.1866353					0.154048				
2	0.228544	0.2064503				0.200226	0.193781			
3	0.2242455	0.2179898	0.2045788			0.192326	0.17616	0.167797		
4	0.2456977	0.2182969	0.2276359	0.2107703		0.209087	0.173654	0.183061	0.167819	
5	0.221009	0.2162649	0.2202855	0.2244157	0.2126472	0.188667	0.164509	0.173582	0.179575	0.164874

Parameters	Substitution Matrix: BL45, BL62,BL100; Gap Open Penalty: 12,14,16; Gap Extension Penalty: 2,3,4									
	λ					K				
Cluster	1	2	3	4	5	1	2	3	4	5
1	0.1368407					0.049183				
2	0.213582	0.1701292				0.159863	0.063276			
3	0.2079975	0.192342	0.154174			0.14941	0.096799	0.040994		
4	0.2373249	0.1928895	0.207206	0.1650268		0.198173	0.095714	0.123883	0.043014	
5	0.1994941	0.1937772	0.1957841	0.2015482	0.1787635	0.12346	0.099895	0.104908	0.10812	0.066458

Parameters	Substitution Matrix: BL45, BL62; Gap Open Penalty: 12,14; Gap Extension Penalty: 2,3									
	λ					K				
Cluster	1	2	3	4	5	1	2	3	4	5
1	0.1402432					0.05231				
2	0.2145139	0.1712422				0.100205	0.045569			
3	0.2093419	0.1982169	0.1540202			0.101861	0.091968	0.031788		
4	0.2411866	0.1931565	0.2063857	0.1642365		0.125784	0.063434	0.075368	0.030885	
5	0.1987875	0.1941648	0.1946307	0.2020014	0.1783505	0.079681	0.064921	0.066602	0.072043	0.046346

Parameters	Substitution Matrix: BL45, BL100; Gap Open Penalty: 12,16; Gap Extension Penalty: 2,4									
	λ					K				
Cluster	1	2	3	4	5	1	2	3	4	5
1	0.1382574					0.05416				
2	0.2141082	0.1717717				0.14858	0.065682			
3	0.2092158	0.1932967	0.1537908			0.141856	0.091108	0.036667		
4	0.2404284	0.1954375	0.2076836	0.1648767		0.208871	0.104467	0.113331	0.041522	
5	0.2003162	0.1949345	0.1968851	0.2041312	0.1789941	0.11604	0.094415	0.096769	0.109758	0.059355

Parameters	Substitution Matrix: BL62, BL100; Gap Open Penalty: 14,16; Gap Extension Penalty: 3,4									
	λ					K				
Cluster	1	2	3	4	5	1	2	3	4	5
1	0.1612023					0.088929				
2	0.2201957	0.1928412				0.185743	0.137453			
3	0.2210159	0.2074159	0.1823947			0.251905	0.157085	0.103554		
4	0.2406616	0.210198	0.2198564	0.1994292		0.212245	0.164609	0.180414	0.144506	
5	0.2090656	0.2068202	0.2068121	0.2145398	0.1982235	0.159461	0.152743	0.142889	0.161868	0.12984

The K and λ estimates in table 3 are for 1/3-bit scaled substitution matrices. For each scoring scheme, there is a wide variation in the estimated K and λ values. For instance, in the first sub-table, λ values range from 0.1358571 to 0.2439074, and K values range from 0.018778 to 0.070919, although all the pairwise alignments of random sequences for getting the K and λ estimates in the

first sub-table were done using the SIM program with BLOSUM45 substitution matrix, gap open penalty 12, and gap extension penalty 2, i.e. using the same scoring scheme. Since the only contributing factor for the difference between K and λ values for different cluster pairs is the amino acid composition, we can observe that the statistical parameters heavily depend on the amino acid composition. Clustering the protein sequences into groups of similar amino acid composition has therefore to some degree separated the dependence of the statistical significance parameters on the amino acid composition, which is very helpful for quick and accurate estimates of statistical significance for specific pairwise alignments. Once parameter estimation for the cluster-pairs is done for a given scoring scheme, subsequent statistical significance estimation for any sequence pair using the same soring scheme is very quick, since it only involves classification of the sequences to corresponding clusters, and using the statistical parameters for the corresponding cluster-pair.

6 Conclusion and Future Work

The implementation of a clustering-classification based approach for estimating the statistical parameters K and λ for estimating the statistical significance of pairwise alignments is done and is experimented with. The clusters are based on the amino-acid composition and the estimates of the statistical parameters K and λ for each cluster-pair are calculated by simulation. Given two sequences, the estimate of K and λ for that pair is given by the K and λ values corresponding to the cluster-pair to which the given sequences are classified based on the amino-acid-composition.

The estimated values of K and λ for different clusters show a considerable variability, even for the same alignment scoring scheme, which suggests that the influence of amino acid composition on statistical parameters K and λ is very strong, and it is imperative to use different K and λ values for different sequences. The clustering technique used in this work has therefore separated the influence of amino acid composition on statistical parameters, which is the main contribution of this paper. Another major significance of this work is that this method can be applied to any new alignment program with any scoring scheme without the knowledge of the statistics of the alignment procedure (which is in general difficult to determine). Once the influence of amino acid composition on statistical significance parameters is separated from other factors, all that needs to be done is the accurate estimation of the statistical parameters for all cluster pairs using the new alignment program, and subsequently use those values for any pair of sequences with individually similar amino acid composition as that of the clusters to which the pair of sequences are individually classified. Especially with a number of new alignment methods being developed, this technique is expected to be very useful in comparing them.

Although the simple idea is very promising, it is unclear how well it works for an application where statistical significance is used, like homology detection. This approach is just a beginning of the efforts to separate the influence of amino

acid composition, and clustering is just one of the many methods which can do so. It may be possible that it is not the exact composition clusters that two protein sequences under comparison fall into that matters, but instead, simply the difference between the composition distributions of the two proteins, which needs further exploration. Another shortcoming of this work is that by clustering hundreds of thousands of sequences in to just five clusters, we lose a lot of information about the amino acid composition distribution across the real protein sequences. An analytical study of the amino acid composition distribution may be required to get the optimal number of clusters. Hence, this method can be further looked into in detail to evaluate the performance of clustering. Another improvement can be to use a small scale simulation along with the proposed approach to increase the accuracy of the statistical significance estimates.

References

1. Altschul, S.F., Madden, T.L., Schäffer, A.A., Zhang, J., Zhang, Z., Miller, W., Lipman, D.J.: Gapped BLAST and PSI-BLAST: A New Generation of Protein Database Search Programs. Nucleic Acids Research 25(17), 3389–3402 (1997)
2. Smith, T.F., Waterman, M.S.: Identification of Common Molecular Subsequences. Journal of Molecular Biology 147(1), 195–197 (1981)
3. Sellers, P.H.: Pattern Recognition in Genetic Sequences by Mismatch Density. Bulletin of Mathematical Biology 46(4), 501–514 (1984)
4. Pearson, W.R.: Effective Protein Sequence Comparison. Methods in Enzymology 266, 227–259 (1996)
5. Pearson, W.R.: Flexible Sequence Similarity Searching with the FASTA3 Program Package. Methods in Molecular Biology 132, 185–219 (2000)
6. Huang, X., Chao, K.M.: A Generalized Global Alignment Algorithm. Bioinformatics 19(2), 228–233 (2003)
7. Huang, X., Brutlag, D.L.: Dynamic Use of Multiple Parameter Sets in Sequence Alignment. Nucleic Acids Research 35(2), 678–686 (2007)
8. Karlin, S., Altschul, S.F.: Methods for Assessing the Statistical Significance of Molecular Sequence Features by Using General Scoring Schemes. Proceedings of the National Academy of Sciences, USA 87(6), 2264–2268 (1990)
9. Pearson, W.R.: Empirical Statistical Estimates for Sequence Similarity Searches. Journal of Molecular Biology 276, 71–84 (1998)
10. Mott, R., Tribe, R.: Approximate Statistics of Gapped Alignments. Journal of Computational Biology 6(1), 91–112 (1999)
11. Mott, R.: Accurate Formula for P-values of Gapped Local Sequence and Profile Alignments. Journal of Molecular Biology 300, 649–659 (2000)
12. Altschul, S.F., Bundschuh, R., Olsen, R., Hwa, T.: The estimation of statistical parameters for local alignment score distributions. Nucleic Acids Research 29(2), 351–361 (2001)
13. Schäffer, A.A., Aravind, L., Madden, T.L., Shavirin, S., Spouge, J.L., Wolf, Y.I., Koonin, E.V., Altschul, S.F.: Improving the Accuracy of PSI-BLAST Protein Database Searches with Composition-based Statistics and Other Refinements. Nucleic Acids Research 29(14), 2994–3005 (2001)
14. Bundschuh, R.: Rapid Significance Estimation in Local Sequence Alignment with Gaps. In: RECOMB 2001: Proceedings of the fifth annual International Conference on Computational biology, pp. 77–85. ACM, New York (2001)

15. Poleksic, A., Danzer, J.F., Hambly, K., Debe, D.A.: Convergent Island Statistics: A Fast Method for Determining Local Alignment Score Significance. Bioinformatics 21(12), 2827–2831 (2005)
16. Kschischo, M., Lässig, M., Yu, Y.: Toward an Accurate Statistics of Gapped Alignments. Bulletin of Mathematical Biology 67, 169–191 (2004)
17. Grossmann, S., Yakir, B.: Large Deviations for Global Maxima of Independent Superadditive Processes with Negative Drift and an Application to Optimal Sequence Alignments. Bernoulli 10(5), 829–845 (2004)
18. Pearson, W.R., Wood, T.C.: Statistical Significance in Biological Sequence Comparison. In: Balding, D.J., Bishop, M., Cannings, C. (eds.) Handbook of Statistical Genetics, pp. 39–66. Wiley, Chichester (2001)
19. Mott, R.: Alignment: Statistical Significance. Encyclopedia of Life Sciences (2005), http://mrw.interscience.wiley.com/emrw/9780470015902/els/article/a0005264/current/abstract
20. Mitrophanov, A.Y., Borodovsky, M.: Statistical Significance in Biological Sequence Analysis. Briefings in Bioinformatics 7(1), 2–24 (2006)
21. Eddy, S.R.: Multiple Alignment Using Hidden Markov Models. In: Rawlings, C., Clark, D., Altman, R., Hunter, L., Lengauer, T., Wodak, S. (eds.) Proceedings of the Third International Conference on Intelligent Systems for Molecular Biology, pp. 114–120. AAAI Press, Menlo Park (1995)
22. Eddy, S.R.: Maximum Likelihood Fitting of Extreme Value Distributions (1997), unpublished manuscript, citeseer.ist.psu.edu/370503.html
23. Agrawal, A., Brendel, V., Huang, X.: Pairwise Statistical Significance Versus Database Statistical Significance for Local Alignment of Protein Sequences. In: Măndoiu, I., Sunderraman, R., Zelikovsky, A. (eds.) ISBRA 2008. LNCS(LNBI), vol. 4983, pp. 50–61. Springer, Heidelberg (in press, 2008)
24. Olsen, R., Bundschuh, R., Hwa, T.: Rapid Assessment of Extremal Statistics for Gapped Local Alignment. In: Proceedings of the Seventh International Conference on Intelligent Systems for Molecular Biology, pp. 211–222. AAAI Press, Menlo Park (1999)
25. Anderson, T.W.: An Introduction to Multivariate Statistical Analysis, 2nd edn. Wiley-Interscience, Chichester (2003)
26. Language, R.A.: Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria (2006)
27. Huang, X., Miller, W.: A Time-efficient Linear-space Local Similarity Algorithm. Advances in Applied Mathematics 12(3), 337–357 (1991)

Gapped Extension for Local Multiple Alignment of Interspersed DNA Repeats

Todd J. Treangen[1,⋆], Aaron E. Darling[2,⋆], Mark A. Ragan[2], and Xavier Messeguer[1]

[1] Dept. of Computer Science, Polytechnic University of Catalonia, Barcelona, Spain
treangen@lsi.upc.edu
[2] ARC Centre of Excellence in Bioinformatics, and Institute for Molecular Bioscience, The University of Queensland, Brisbane, Australia
a.darling@imb.uq.edu.au

Abstract. The identification of homologous DNA is a fundamental building block of comparative genomic and molecular evolution studies. To date, pairwise local sequence alignment methods have been the prevailing technique to identify homologous nucleotides. However, existing methods that identify and align all homologous nucleotides in one or more genomes have suffered poor scalability and limited accuracy. We propose a novel method that couples a gapped extension heuristic with a previously described efficient filtration method for local multiple alignment. During gapped extension, we use the MUSCLE implementation of progressive multiple alignment with iterative refinement. The resulting gapped extensions potentially contain alignments of unrelated sequence. We detect and remove such undesirable alignments using a hidden Markov model to predict the posterior probability of homology. The HMM emission frequencies for nucleotide substitutions can be derived from any strand/species-symmetric nucleotide substitution matrix, and we have developed a method to adapt an arbitrary substitution matrix (i.e. HOXD) to organisms with different G+C content. We evaluate the performance of our method and previous approaches on a hybrid dataset of real genomic DNA with simulated interspersed repeats. Our method outperforms existing methods in terms of sensitivity, positive predictive value, and localizing boundaries of homology. The described methods have been implemented in the free, open-source `procrastAligner` software, available from: `http://alggen.lsi.upc.es/recerca/align/procrastination`

1 Introduction

The importance of accurate homology identification to comparative genomics cannot be overestimated[1]. To date, pairwise local sequence alignment methods such as BLAST [2,3,4] have been the prevailing technique to identify homologous nucleotides. When more than two copies of a homologous sequence element are present in the data, pairwise homology detection methods generate a listing of all possible pairs of homologous elements in the form of pairwise local alignments. Apart from the obvious inefficiency of considering all pairwise homology relationships, a collection of pairwise

⋆ These authors contributed equally to this work.

I. Măndoiu, R. Sunderraman, and A. Zelikovsky (Eds.): ISBRA 2008, LNBI 4983, pp. 74–86, 2008.

alignments is not ideal because they are rarely amenable to comparative genomic and phylogenetic analysis without further processing into a multiple alignment.

Local pairwise alignments can be merged to create a multiple alignment by a variety of methods[5,6,7,8]. Such methods commonly assume that pairwise homology relationships are transitive, such that if nucleotide a is homologous to nucleotide b, and b is to c, then a must also be homologous to c. Thus, in order to merge pairwise alignments, such methods must tackle the challenging problem of resolving inconsistent transitive homology relationships. Multiple alignment has been demonstrated to be more accurate than pairwise alignment, especially when dealing with a large number of divergent sequences[9,10]. As the number of homologous sequences grows, we might expect that the number of inconsistent relationships in a collection of pairwise alignments would grow quadratically, whereas a direct multiple alignment method would provide an increasingly accurate alignment. Moreover, highly repetitive regions in the input sequences can cause serious efficiency problems for pairwise methods, as they create $O(r^2)$ pairwise alignments in the presence of a repeat with r copies. Mammalian Alu repeats and IS elements in microbes are two common examples of the overwhelming abundance of repetitive sequence in whole genomes.

Local multiple alignment has the inherent potential to avoid pitfalls associated with pairwise alignment. Although optimal multiple alignment under the SP objective function remains intractable[11], progressive alignment heuristics offer excellent speed and accuracy[12,13] especially when combined with tree-independent iterative refinement [14], or probabilistic consistency measures[15]. Rather than merging pairwise alignments, why not exploit years of research into multiple alignment heuristics by directly constructing a multiple alignment? We thus present a novel heuristic for directly computing local multiple alignments via gapped extension of chained seed matches.

2 A Heuristic for Gapped Extension of Local Multiple Alignments

Our method for computing local multiple alignments exploits the MUSCLE multiple alignment algorithm to compute gapped extensions of ungapped multi-match seeds (see Fig. 1). Gapped alignments arise when extending seeds to fully capture surrounding sequence homology. Our method assumes that a fixed number of nucleotides flanking a seed match are likely to be homologous and computes a global multiple alignment on the flanking region. Our assumption of flanking homology often proves to be erroneous and results in an alignment of unrelated sequences. In the context of *local* multiple alignment, the fundamental problem with such an approach is that current methods for progressive alignment with iterative refinement compute *global* alignments, i.e. they implicitly assume that input sequences are homologous over their entire length. To resolve the problem, we employ a hidden Markov model which detects unrelated regions embedded in the global multiple alignment. Unrelated regions are then removed from the alignment and the local multiple alignment is trimmed to reflect the updated boundaries of homology.

Our method, depicted for an example sequence in Fig. 1, has seven primary steps: (1) identify multi-match seeds in the input sequence, (2) chain individual seeds, (3)

multiply align of regions between chained seeds, (4) gapped extension of seed chains (5) detect unrelated regions using a hidden Markov model, (6) apply transitive homology relationships, and (7) removal of any unrelated sequence from the final local multiple alignment. We have previously published Steps 1-2 of our method[16], while steps 3-7 represent a new contribution and are the subject of the present manuscript. Steps 2-7 are applied repeatedly to seeds identified in step 1 to produce local multiple alignments of all homologous nucleotides in the input sequence.

2.1 Chaining Multi-match Seeds from the Input Sequence

Given a sequence $\mathcal{S} = s_1, s_2, \ldots, s_N$ of length N defined over an alphabet $\{A, C, G, T\}$, our method identifies local multiple alignments on homologous subsequences of $\mathcal{S}$. Our method first extracts candidate ungapped alignments, or multi-matches, among subsequences in $\mathcal{S}$, and we denote the set of all such matches as $\mathbf{M}$. To extract multi-matches from the input sequence, we utilize a palindromic spaced seed pattern[17], which is analyzed at each position in the input sequence. Palindromic spaced seeds offer good efficiency and reasonable sensitivity on a variety of input sequences[16]. We refer the number of matching regions in $\mathcal{S}$ by a given match $M_i \in \mathbf{M}$ as the *multiplicity* of M_i, denoted as $|M_i|$. We refer to each matching region of M_i as a *component* of M_i. Our algorithm has an important limitation on the matches in $\mathbf{M}$: no two matches M_i and M_j may have the same left-end coordinate, except for the identity case when $i = j$. This constraint has been referred to by others as *consistency* and *transitivity*[18] of matches.

Once a list of multi-matches has been generated, we employ an efficient chaining and filtration algorithm to identify overlapping and nested chains of multi-matches[16]. In order to process each region of sequence $\mathcal{O}(1)$ times, matches are prioritized for chaining in order of decreasing multiplicity. The method chains multi-match seeds of the same multiplicity $|M_i|$ occurring within w characters of each other, thus gaps of up to size w are tolerated. When a multi-match can no longer be chained without including a gap larger than w characters, neighboring *subset* matches within w characters are identified. Each neighboring subset match is then *linked* to the chained match. We refer to the chained match as a *superset* match. Rather than immediately extend the subset match(es), we *procrastinate* and extend the subset match later when it has the highest multiplicity of any match waiting to be extended. When chaining a match with a linked superset, we immediately include the entire region covered by the linked superset match and thus eliminate the need to re-examine sequence already covered by a previously chained match.

2.2 Gapped Extension of High-Scoring Chains

Our new method computes gapped extensions of the chained multi-match seeds. After chaining a multi-match M_i, we perform gapped alignment on all collinear regions located between two adjacent components to generate unextended local multiple alignments. We first evaluate the chain to decide whether expending computational resources on gapped extension will be worthwhile. We can optionally require that two or more seeds be present in the chain and use lower seed weights (k), a technique which

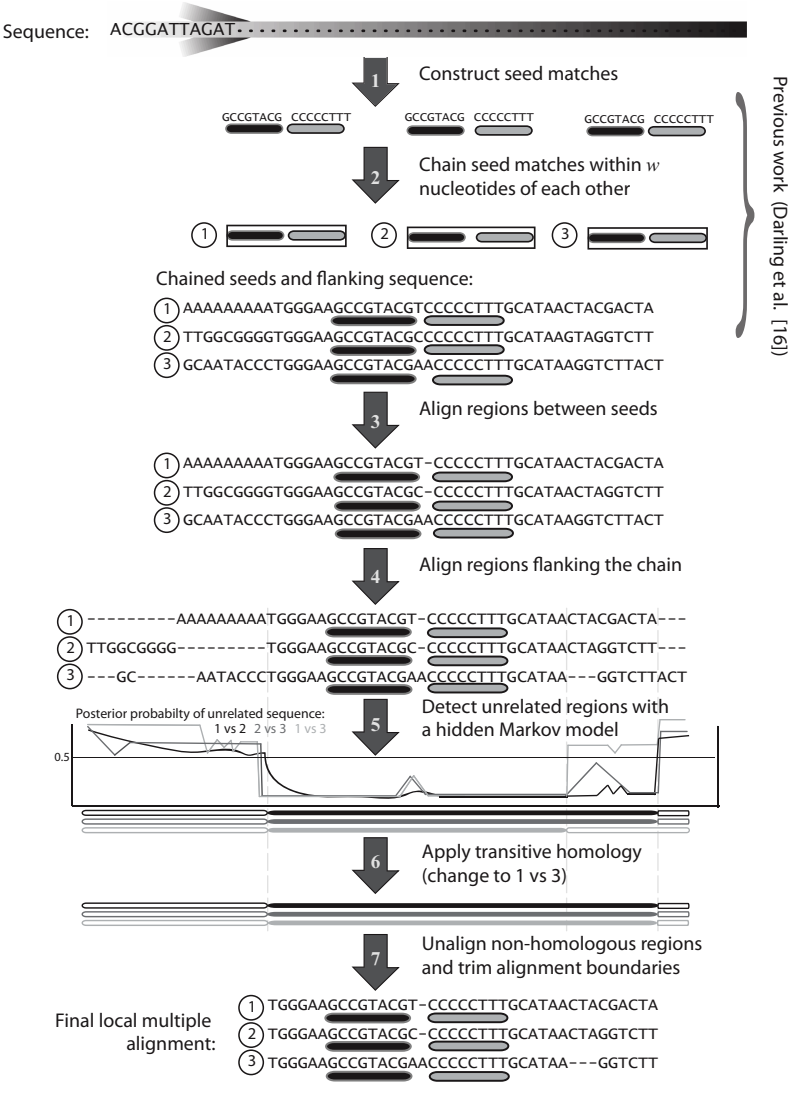


Fig. 1. Overview of the method, starting with an input sequence and ending with a set of local multiple alignments. First we (1) detect multi-matches in the input sequence(s) using palindromic spaced seeds, then we perform (2) chaining and extension of all multi-matches within w nt of each other. In the present example, one chain exists and contains two matches each with three match components labeled 1, 2, and 3. We then perform gapped alignment of the region between chained matches (3). In step (4), we perform a gapped extension by computing a global multiple alignment on the regions to the left and right of each chain component. The resulting alignment may contain unrelated sequence, so in step (5) we apply a hidden Markov model to detect poorly aligned regions indicative of unrelated sequence. Step (6) computes transitive homology relationships to ensure a consistent alignment and aid detection of divergent homologous sequences. Finally, in step (7) we unalign regions found to be non-homologous. If we find after step (2) that the alignment boundaries have been extended, we return to step (4) for another round of chaining.

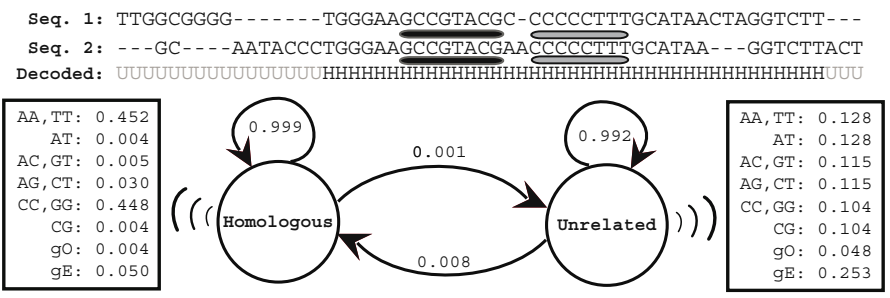


Fig. 2. Hidden Markov model used to detect pairwise alignments of unrelated sequence. The HMM has states which model alignment columns containing homologous and unrelated sequence. Emission probabilities are extracted from the HOXD substitution matrix and correspond to alignment columns, for example AA indicates A aligned to A. gO indicates gap-open and gE gap extend. Alignment columns are treated as strand-symmetric, so that AC also indicates CA and the reverse complements TG and GT. The emission probabilities are adjusted to the G+C content of the input genome as described in the test. The values shown here correspond to a 47.5% G+C genome.

has previously been proven successful[2,19,20]. To perform a gapped extension in each direction, we use MUSCLE to align dynamically-calculated window of nucleotides (l) to the left and right of the current local multiple alignment. Small values of l restrict the alignment search space, while larger values require more computation but are potentially more sensitive. We have empirically determined that setting l based on multiplicity ($l = 70e^{-0.01*|M_i|}$) offers a good tradeoff between speed and sensitivity. The resulting extension window is small for high multiplicity chains, keeping the alignment search space tractable.

2.3 Identifying Unrelated Regions

The MUSCLE alignment software dutifully reports the highest scoring global multiple alignment of input sequences, regardless of whether they are related by common ancestry. As a consequence of the gapped extension process, it is likely that our method forcibly aligns unrelated sequence. We have configured a hidden Markov model (Fig. 2) to detect alignments of unrelated sequence. The HMM consists of two hidden states, Homologous and Unrelated. The observable states are the pairwise alignment columns, which are all possible pairs in {A,G,C,T,-} with strand and species symmetry, i.e. AG=GA=TC=CT. The emission probabilities for each possible pair of aligned nucleotides were extracted from the HOXD substitution matrix presented by Chiaromonte *et al.* [21]. We solved for the emission frequencies in the homologous and unrelated state using the same equation used to calculate the values of the HOXD substitution matrix on 47.5%G+C content sequence[21]:

$$s(x,y) = \log_2 \left(\frac{p(x,y)}{q_1(x)q_2(y)} \right) \quad (1)$$

where $p(x, y)$ is the fraction of the observed aligned pairs of nucleotides x and y in the training set used and $q_1(x)$ and $q_2(y)$ denote the background frequencies of x and y, respectively. Chiaromonte *et al.* scaled the resulting $s(x, y)$ values by $\psi = 32.5421$ so the largest was 100, with the rest rounded to the nearest integer. The resulting emission probabilities for the Homologous and Unrelated states are given in Fig. 2. HMM probabilities can be derived using any strand/species-symmetric nucleotide substitution matrix, but any particular matrix makes specific assumptions about divergence time, mutation pressures, and sequence composition of the aligned sequences. Genomes can range in G+C content from 30-75%, and at the extremes, a substitution matrix derived on 47.5% G+C sequence (such as HOXD) does not perform well. Previously it has been shown that adapting substitution matrices to the composition of the organisms under comparison can improve sequence alignment accuracy[22]. We have thus developed a method to adapt HMM emission frequencies derived from an arbitrary substitution matrix to organisms with different G+C content (see `http://alggen.lsi.upc.es/recerca/align/procrastination/` for details).

While emission frequencies for nucleotide substitutions can be derived from any strand/species-symmetric nucleotide substitution matrix, the gap-open and extend frequencies can not. To empirically estimate gap-open and extend values for the unrelated state we aligned a 10-kb, 48% G+C content region taken from *E. coli* CFT073 (Accession AF447814.1, coordinates 37,300-38,300) with an unrelated sequence. We generated an unrelated sequence with identical nucleotide composition by reversing the extracted sequence without complementation. We then forced an alignment with MUSCLE and counted the number of gap-open and gap-extend columns in the alignment of unrelated sequences. Gap-open and extend frequencies for the homologous state were estimated by constructing an alignment of 10kb of orthologous sequence shared among a pair of divergent organisms. We aligned the 48%G+C segment between genes *fruR* and *secA* from *E. coli* K12 (Accession U00096.21) and *Y. pestis* CO92 (Accession AL590842.1). We add the empirically derived gap-open and extend frequencies for each state and normalize the emission frequencies to a probability distribution. The resulting emission probabilities are reported in Fig. 2.

Using the empirically derived transition and emission probabilities, we apply the posterior HMM decoder implemented in the HMMoC software[23] to compute the posterior probability (p.p.) that each alignment column represents homologous sequence. Columns with a p.p. below 0.5 are considered to be unrelated; use of 0.5 as a p.p. threshold yields a maximum *a posteriori* estimate of homology. We then apply the transitive homology principle to our predictions, resulting in a final set of consistent homology predictions. See Fig. 1, steps 5 and 6 for an example. We trim the alignment to exclude all columns beyond the Homologous state. If the original boundaries were improved, we trigger another round of chaining (and consequently another round of extension) in the same direction. When gapped extension fails to improve boundaries in one direction, extension in the other direction is attempted until no further extension is possible.

3 Results

We have previously demonstrated the sensitivity of our chaining method in finding Alu repeats in the human genome[16]. Figure 6 shows part of a local multiple alignment of one such Alu family as generated with `procrastAligner`. To highlight the benefits of our proposed heuristic for gapped extension, we compare `procrastAligner`'s performance to the Eulerian path method for local multiple alignment as implemented by `eulerAlign`[8]. The Eulerian path method uses a *de Bruijn* graph for filtration, and goes beyond filtration to compute gapped alignments using banded dynamic programming. To our knowledge, `procrastAligner` and `eulerAlign` represent the only two automated methods to construct local multiple alignments directly from genomic DNA.

3.1 Simulating Interspersed Repeats

We evaluate accuracy of each method when aligning simulated repeat families that have been inserted into the complete genome of *Mycoplasma genitalium*. The *M. genitalium* genome has been recognized as complex and repeat-rich[24], presenting a biologically relevant and challenging example to evaluate alignment methods. We simulated repeat families of 8 different multiplicities ranging between 2 and 256 (x-axis in Fig. 3). Each repeat copy has an average length based on its family's multiplicity ($length = \frac{7680}{multiplicity}$), thus high copy-number repeats are short. Evolution of repeat families was simulated as a marked Poisson process on a star tree topology. The branch lengths were varied between 0 and 24 (y-axis in Fig. 3), with the nucleotide substitution rate fixed at 0.09 per unit time, and the indel rate fixed at 0.01 per unit time. Rate heterogeneity among sites was modeled with a gamma distribution ($\theta = 1.0, k = 0.5$). Indel size was Poisson

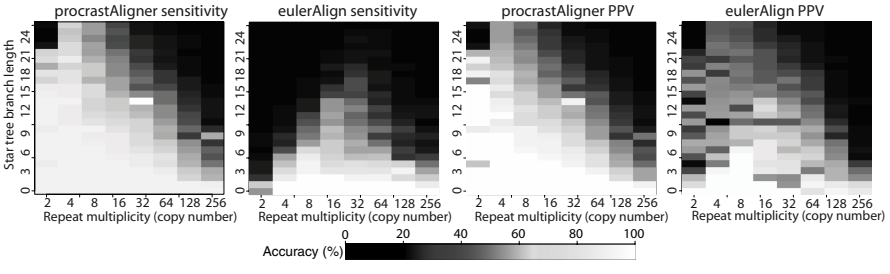


Fig. 3. Accuracy recovering simulated repeat families planted in the *Mycoplasma genitalium* genome. Sum-of-pairs (SP) nucleotide sensitivity and positive predictive value (PPV) of `procrastAligner` and `eulerAlign` were measured for 200 combinations of branch length and multiplicity. Three replicates of each simulation were performed and average accuracy values are shown here. White points indicate perfect alignment of the simulated repeat family. Black points indicate the program completely failed to recover any portion of the repeat family. Average mutations per site can be calculated by multiplying branch length by the fixed substitution rate of 0.09, and indel rate of 0.01. For example, at branch length 20 there are 1.8 substitutions per site and 0.2 indels per site. From the figure, it is apparent that `procrastAligner` performs better at higher mutation rates and multiplicities than `eulerAlign`.

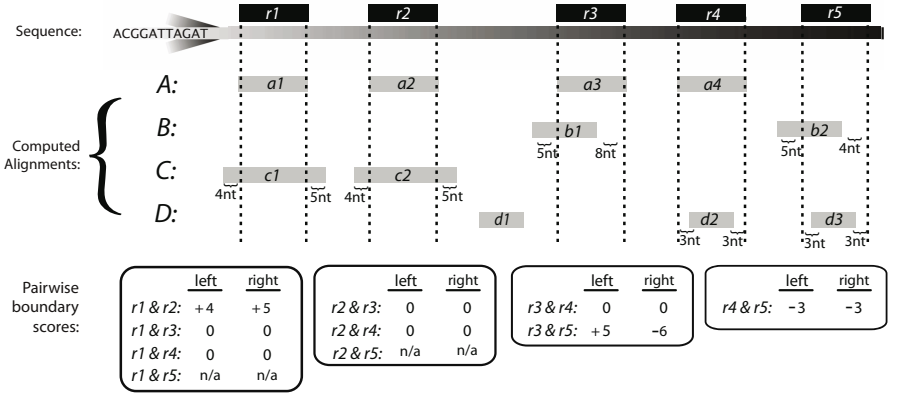


Fig. 4. Pairwise boundary accuracy metric. We define our boundary accuracy metric to be an all-pairs score by comparing the boundaries of all *r* components of the inserted repeat *R* to the boundaries predicted by the alignment program. For any pair of components, r_i and r_j, we take the maximum boundary of any local multiple alignments output by the program. The figure shows a multiplicity five interspersed repeat *R* and four local multiple alignments, *A, B, C, D*. Boundary predictions can be classified as (1) correct, (2) overprediction, and (3) underprediction, with each discussed in turn: (1) *Correct prediction*. Consider scoring components *r1* and *r3*. Local multiple alignment *A* overlaps both *r1* and *r3* and no other alignment overlaps both *r1* and *r3*. The left and right boundaries of alignment *A* match the boundaries of *r1&r3* exactly, thus we assign scores of 0 for *r1&r3*. (2) *Overprediction*. Consider scoring components *r1* and *r2*. These components are overlapped by alignments *A* and *C*. Alignment *A* has perfect boundary predictions for *r1&r2*, while alignment *C* extends beyond the true boundaries of components *r1* and *r2* by 4 nucleotides on the left and 5 nucleotides on the right. Our scoring metric always uses the maximum predicted boundaries for a pair of components, thus the boundary predictions from *C* are reported for *r1&r2*. (3) *Underprediction.* Consider scoring components *r3* and *r5*. Alignment *B* hits both *r3* and *r5*, but stops short of the right-side boundary by 8nt in *r3* and 4nt in *r5*. We average the error and record -6 for the right-side of *r3&r5*. Finally, component pairs that are not contained by any computed alignments are not scored, as indicated by *n/a*.

distributed with intensity 3, and insertions and deletions were taken to be equally likely. Each family's ancestral sequence was randomly generated using nucleotide frequencies equal to the composition of *Mycoplasma genitalium* ($A = 0.34, T = 0.34, G = 0.16, C = 0.16$). Insertion sites for repeat copies were chosen uniformly at random in the 580kb *M. genitalium* genome, allowing tandem repeats but prohibiting mosaic repeats.

3.2 Alignment Accuracy Metrics

We used each program to find local multiple alignments in each of the 200 modified *M. genitalium* genomes and recorded alignment accuracy as follows. We calculated Sum-of-Pairs (SP) nucleotide sensitivity as $\frac{\mathrm{TP}}{\mathrm{TP+FN}}$, where TP is the number of aligned nucleotide pairs in the program's output which are also aligned in the simulated repeat family. FN is the number of aligned nucleotide pairs in the simulated repeat family which are missing from the program's output. This sensitivity measure is identical to

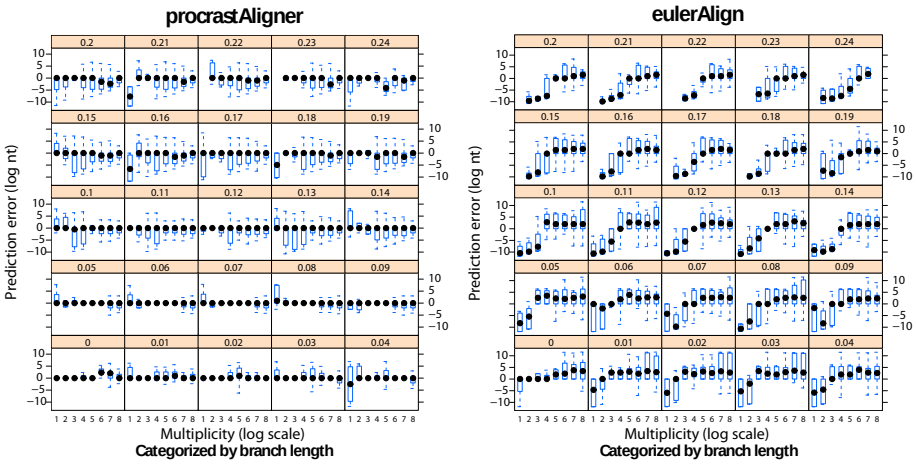


Fig. 5. Boundary prediction performance. All-pairs boundary prediction accuracy of `procrastAligner` and `eulerAlign` were measured for 200 combinations of branch length and multiplicity. Accuracy on each combination is presented as a box-and-whiskers plot using the scoring metric detailed in Section 3.2. Branch lengths range from 0 to 0.24 and increase by intervals of 0.01. The x-axis label represents the multiplicity of the interspersed repeat in $\log_2$-scale. i.e. axis label 8 indicates 2^8 = multiplicity 256. The y-axis label is the prediction error in $\log_2$-scale nucleotides. Values at 0 represent correctly identified repeat boundaries, values greater than 0 represent overpredictions, and values less than 0 represent underpredictions (see Fig. 4). In general, `procrastAligner` identifies the true interspersed repeat boundaries more accurately than `eulerAlign`.

the Sum-of-Pairs (SP) accuracy defined by BaliBASE[25]. We calculate the positive predictive value (PPV) as $\frac{\mathrm{TP}}{\mathrm{TP+FP}}$, where TP is defined as above, and FP is the total number of nucleotide pairs from the program's output where one of the nucleotides are part of the simulated repeat family and the other nucleotide was incorrectly aligned. We also quantify the ability of each aligner to accurately predict the boundaries of the interspersed repeats. For a given pair of repeat components, we calculate accuracy by recording the number of nucleotides between the true boundary and the predicted boundary on both the right and left sides of the repeat. Thus, over-extension gets a positive score, while underextension yields a negative score and perfect boundaries receive a 0 score. See Fig. 4 for further details on boundary under/overpredictions.

3.3 Accuracy When Aligning Interspersed Repeats

We applied `procrastAligner` and `eulerAlign` to the hybrid simulated & real dataset. We ran `procrastAligner` with command-line parameters `--z=15 --w=20` and `eulerAlign` with `-k 15 -l -i 1000 -v` based on suggestions from the program's user guide and manual experimentation. Simulations for each of the 200 combinations of branch length and multiplicity were replicated three times and alignments generated in parallel on a 156-node compute cluster. Results of the

experiments are reported in Fig. 3 and Fig. 5. Figure 3 illustrates the sensitivity and PPV of both methods on datasets ranging from 0 substitutions and indels per site to 2.16 substitutions and 0.24 indels per site (branch length 24). As mutation rates and repeat multiplicity increase the alignment accuracy decreases for both methods, with accuracy of `eulerAlign` decreasing faster than `procrastAligner`. Surprisingly, `eulerAlign` often fails to align low multiplicity repeats, even when mutation rates are low. Manual experimentation with `eulerAligner` parameters, such as: -v (tolerance for mismatches), -k (seed k-mer size) from 11 to 15, and -i (number of iterations) from 1000 up to 10,000, failed to improve its performance on low-multiplicity repeats. We conjecture that `procrastAligner`'s overall improved accuracy largely derives from its use of spaced seed patterns[16] and tolerance of gaps. With the `-v` option enabled, the Eulerian path method allows up to 10% mismatches for matching k-mers to seed gapped alignment extensions. While this certainly improves the sensitivity at lower mutation rates, the experimental results presented Fig. 3 show that it is inadequate for higher mutation rates.In addition to sensitivity and PPV benchmarks, we also assess how well each aligner recovers the true boundaries of interspersed repeats. Figure 5 illustrates the ability of each program to accurately localize the known boundaries of the simulated interspersed repeats. From the figure, it is apparent that on average, `procrastAligner` predicts the exact repeat boundary for all studied combinations of branch length (repeat degeneracy) and multiplicity (repeat copy number). Moreover, the standard error in `procrastAligner`'s boundary predictions is typically very low, within 4 nucleotides. `eulerAlign`, on the other hand, exhibits more erratic behavior. For low multiplicity repeats it has a strong tendency to underpredict the repeat boundary. At high multiplicities ($\geq$ 32) `eulerAlign` tends to slightly overpredict the boundaries, by including flanking unrelated sequence in the alignment of the interspersed repeat.

Finally, a direct comparison of run time is difficult, due to the differing natures of the local multiple alignment programs. `eulerAlign` runtime depends on the iterations parameter, which controls the total number of local multiple alignments reported, whereas `procrastAligner` reports *all* local multiple alignments in a single run. Despite this, we report the average per-experiment CPU time for each program on the test dataset. `procrastAligner` required on average 55 seconds per experiment, with the longest taking just over two minutes. `eulerAlign` required 1 hour total compute time per experiment on average, which equates to about 4 seconds per iteration. Both programs exhibited similar memory usage, `procrastAligner` requiring on average 50 MB per experiment and `eulerAlign` requiring on average 70 MB per experiment.

4 Discussion

We have presented a sensitive and efficient gapped-extension heuristic for local multiple alignment. We have extended our previous results by converting chains of ungapped multi-matches into gapped local multiple alignments. Our method is based around an efficient heuristic for local multiple alignment, featuring a novel method for gapped extensions joining global multiple alignment with a homology test based on a hidden Markov model. Experimental results demonstrate that the described method offers a

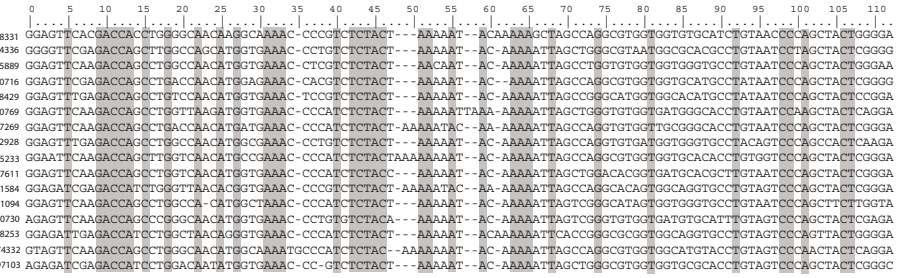


Fig. 6. Alu repeat alignment. Partial view of an Alu repeat alignment output by procrastAligner in the *H. sapiens* BAC clone RP11-355H10 (Accession AC010145.10). Each row represents an aligned Alu. Highlighted columns indicate conserved sequence among all 16 copies of the Alu. Start positions are shown to the left, negative values indicate complement strand. Local multiple alignment was generated with `procrastAligner` with parameters: `--z=9 --w=50`.

level of alignment accuracy exceeding that of previous methods. Accurately predicting homology boundaries has important implications; for example, tools to build repeat family databases can directly use the alignments without the manual curation required in current approaches and also is likely to aid in the evolutionary analysis of transposon proliferation. Further improvement of the alignment methodology will likely require increasingly sensitive methods for seed matching in conjunction with a statistical methodology to assign significance to local multiple alignments. One possible avenue to increase seed matching sensitivity and reduce boundary underpredictions would be merging overlapping seed matches into a shorter, higher-multiplicity match. A second avenue would be to use of palindromic seed families instead of using a single seed pattern. With increased seed matching sensitivity comes additional false positive seed hits, so a statistical test for rejecting insignificant local alignments will likely be required. Unfortunately exact computation of p-values for local multiple alignments remains a daunting challenge, although fast approximation methods for pairwise alignments have shown promise[26] and potentially can be extended to multiple alignments[8,27].

4.1 Implementation

We have implemented our method in a program, `procrastAligner`, available for Linux, Windows, and Mac OS X. Our open-source implementation is available as C++ source code licensed under the GPL , and can be downloaded from:
`http://alggen.lsi.upc.es/recerca/align/procrastination`.

Acknowledgments

The authors would like to thank Yu Zhang for providing the `eulerAlign` program. We are grateful to Guillaume Achaz for helpful discussions on the gapped extension algorithm. Accuracy evaluations utilized a compute resource grant from the Australian Partnership for Advanced Computing. AED was supported by NSF grant DBI-0630765.

TJT was supported by Spanish Ministry MECD Grant TIN2004-03382 and AGAUR Training Grant FI-IQUC-2005.

References

1. Kumar, S., Filipski, A.: Multiple sequence alignment: In pursuit of homologous DNA positions. Genome Res. 17, 127–135 (2007)
2. Schwartz, S., Kent, J.W., Smit, A., Zhang, Z., Baertsch, R., Hardison, R.C., Haussler, D., Miller, W.: Human-mouse alignments with blastz. Genome Res. 13, 103–107 (2003)
3. Pearson, W.R.: Rapid and sensitive sequence comparison with FASTP and FASTA. Methods Enzymol 183, 63–98 (1990)
4. Ma, B., Tromp, J., Li, M.: PatternHunter: faster and more sensitive homology search. Bioinformatics 18, 440–445 (2002)
5. Blanchette, M., Kent, W., Riemer, C., Elnitski, L., Smit, A.F., Roskin, K.M., Baertsch, R., Rosenbloom, K., Clawson, H., Green, E., Haussler, D., Miller, W.: Aligning multiple genomic sequences with the threaded blockset aligner. Genome Res. 14, 708–715 (2004)
6. Raphael, B., Zhi, D., Tang, H., Pevzner, P.: A novel method for multiple alignment of sequences with repeated and shuffled elements. Genome Res. 14(11), 2336–2346 (2004)
7. Morgenstern, B., French, K., Dress, A., Werner, T.: DIALIGN: finding local similarities by multiple sequence alignment. Bioinformatics 14, 290–294 (1998)
8. Zhang, Y., Waterman, M.S.: An Eulerian path approach to local multiple alignment for DNA sequences. PNAS 102, 1285–1290 (2005)
9. Brudno, M., Do, D.C.B., Cooper, G.M., Kim, M.F., Davydov, E., Program, N.C.S., Green, E.D., Sidow, A., Batzoglou, S.: LAGAN and Multi-LAGAN: efficient tools for large-scale multiple alignment of genomic dna. Genome Res. 13, 721–731 (2003)
10. Szklarczyk, R., Heringa, J.: Aubergene–a sensitive genome alignment tool. Bioinformatics 22, 1431–1436 (2006)
11. Wang, L., Jiang, T.: On the complexity of multiple sequence alignment. J. Comput. Biol. 1, 337–348 (1994)
12. Thompson, J.D., Higgins, D.G., Gibson, T.: CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position specific gap penalties and weight matrix choice. Nucleic Acids Res. 22, 4673–4680 (1994)
13. Notredame, C., Higgins, D.G., Heringa, J.: T-Coffee: A novel method for fast and accurate multiple sequence alignment. J. Mol. Biol. 302, 205–217 (2000)
14. Edgar, R.: MUSCLE: Multiple sequence alignment with high accuracy and high throughput. Nucleic Acids Res. 32 (2004)
15. Do, C.B., Mahabhashyam, M.S., Brudno, M., Batzoglou, S.: ProbCons: Probabilistic consistency-based multiple sequence alignment. Genome Res. 15, 330–340 (2005)
16. Darling, A.E., Treangen, T.J., Zhang, L., Kuiken, C., Messeguer, X., Perna, N.T.: Procrastination leads to efficient filtration for local multiple alignment. Algorithms in Bioinformatics 4175, 126–137 (2006)
17. Choi, P.K., Zeng, F., Zhang, L.: Good spaced seeds for homology search. Bioinformatics 20, 1053–1059 (2004)
18. Szklarczyk, R., Heringa, J.: Tracking repeats using significance and transitivity. Bioinformatics 20 (suppl. 1), 1311–1317 (2004)
19. Altschul, S.F., Madden, T.L., Schäffer, A.A., Zhang, J., Zhang, Z., Miller, W., Lipman, D.J.: Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. Nucleic Acids Res. 25, 3389–3402 (1997)
20. Kent, W.J.: BLAT–the BLAST-like alignment tool. Genome Res. 12, 656–664 (2002)

21. Chiaromonte, F., Yap, V.B., Miller, W.: Scoring pairwise genomic sequence alignments. In: Pac Symp. Biocomput., pp. 115–126 (2002)
22. Yi-Kuo, Y., Altschul, F.: The construction of amino acid substitution matrices for the comparison of proteins with non-standard compositions. Bioinformatics 21, 902–911 (2005)
23. Lunter, G.: HMMoC a compiler for hidden Markov models. Bioinformatics 23, 2485–2487 (2007)
24. Rocha, E.P., Blanchard, A.: Genomic repeats, genome plasticity and the dynamics of *Mycoplasma* evolution. Nucleic Acids Res. 30, 2031–2042 (2002)
25. Thompson, J.D., Plewniak, F., Poch, O.: A comprehensive comparison of multiple sequence alignment programs. Nucleic Acids Res 27, 2682–2690 (1999)
26. Achaz, G., Boyer, F., Rocha, E.P.C., Viari, A., Coissac, E.: Repseek, a tool to retrieve approximate repeats from large dna sequences. Bioinformatics (2006)
27. Prakash, A., Tompa, M.: Statistics of local multiple alignments. Bioinformatics 21(suppl. 1) (2005)

Improved Alignment of Protein Sequences Based on Common Parts

David Hoksza

Department of software engineering, Faculty of Mathematics and Physics,
Charles University in Prague
Malostranské nám. 25, 118 00, Prague 1, Czech Republic
david.hoksza@mff.cuni.cz

Abstract. In the last twenty years, protein databases have been growing exponentially. To speed up the search, heuristic approaches have been proposed and their accuracy has been steadily growing, but exact search is still needed in some cases. The only exact search algorithm remains SSEARCH (or it's clones) which sequentially scans database of protein sequences, and performs full alignment against each of the sequences.

Due to the need of the exact search, we focus on improving the sequential search algorithm. We decrease the costs needed to compute the alignment of pair of protein sequences when used with large databases. This is achieved by reusing alignment calculations of common parts of the sequences without loss of accuracy.

With this method, we reduced the computational costs by up to 20 % depending on the database size and subset used. We also implemented approximate search which further reduced computational costs for the the sake of some accuracy loss.

Keywords: protein databases, Smith-Waterman algorithm.

1 Introduction

In recent years, there has been an exponential growth of databases of protein sequences. One of the reasons for this growth is the fact that similarity between a protein sequence with an unknown function and a sequences in a database of protein sequences with known functions is used by biologists to help to determine the function of the inspected protein. Also protein sequences of different species can help (if they are sufficiently similar) in getting information about the new protein, and therefore it makes sense to keep at disposal as large repositories of sequences as possible, hence size of the repositories exponentially grows. This growth has an obvious consequence - searching the repositories becomes slower, especially when *Smith-Waterman* (SW) algorithm [16] (which is the most sensitive method for homology search) is used as the similarity measure. The *Smith-Waterman* algorithm is of quadratical complexity, and therefore use of this algorithm became unacceptable for large repositories. To decrease

I. Măndoiu, R. Sunderraman, and A. Zelikovsky (Eds.): ISBRA 2008, LNBI 4983, pp. 87–99, 2008.

quadratic complexity of the search to linear, heuristic algorithms were proposed - most well known are *FASTA* [10] and *BLAST* [1][1].

Similarity of two sequences, when measured with Smith-Waterman algorithm, is called Smith-Waterman (SW) score. SW-score is applied only to a single pair of sequences and thus E-value (expected value computed out of SW-score) was proposed to incorporate some statistics into the output such as lengths of the aligned sequences, size of the database, etc. E-value for a query sequence q, a database sequence and a SW-score S expresses number of database sequences that will score with q equivalently or better than S with respect to the database size. Hence, lower E-value leads to more significant matches. In usual scenario user inputs an E-value and a protein sequence and gets on the output sequences that are similar to the given sequence with E-value lower than the given value[2]. Due to the rigorous nature of the algorithm, Smith-Waterman finds more distant matches than BLAST or FASTA does, thus SW can output also sequences that would have sufficient E-value but BLAST of FASTA would miss it. And there are still situations where accuracy is preferred over speed, e.g. database curation, finding structures by sequence alignment, etc. In these cases, rigorous algorithms are used.

Smith-Waterman algorithm is nowadays implemented in *ScanPS* [2] (original SW algorithm), *SSEARCH* (SW with SWAT optimizations [6]), or *MPsrch* (*MPsrch* is parallelized version of the true Smith and Waterman[3]).

2 Similarity in Sequence Databases

Since protein sequence is a linear sequence of letters upon alphabet of 20 amino-acids[4], algorithms and similarity measures for searching protein databases are similar to algorithms for searching databases of ordinary strings with the difference that the similarity measure used is a bit more complex (because of semantics of protein sequences). As for every database of objects we must foremost define a similarity measure which we will use to compare objects, and based on this measure we can define methods for searching it. In string databases, similarity measure is based on alignment of sequences (usually pairs of sequences[5]). Alignment of sequences is such an arrangement of it's letters (by inserting gaps to each of them) which holds order of the letters. Hence the simplest alignment is defined upon sequences of equal lengths (pairing letters at equal positions). Score of the alignment is number of positions where the sequences differ. Such a scoring system is called *Hamming distance* (Fig. 1a).

[1] Nowadays BLAST and it's clones are the most exploited heuristic algorithms for homology search among protein sequences.

[2] Similar sequences have low E-value but high SW-score.

[3] MPsrch, SSEARCH and ScanPS are operated by EBI and currently handle about 1100 jobs per month as stated by EBI support.

[4] Each of the amino-acids is coded by a triplet of nucleotides from DNA called codons.

[5] Techniques of multiple alignments of sequences are also widely investigated but it's not subject of this paper.

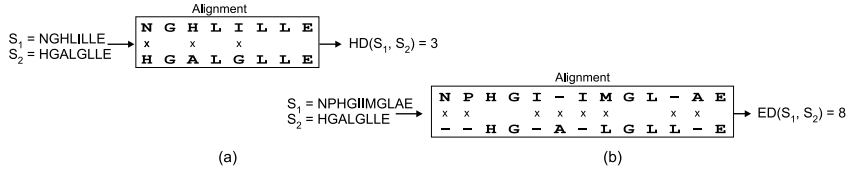


Fig. 1. Hamming (a) and Edit (b) distance

To allow to compare sequences of different lengths we need to insert gaps to them. The desired alignment minimizes number of gaps and positions at which the sequences differ (score of the alignment). Equivalently we can compute minimal number of editing operations needed to turn one sequence into the other where editing operations are inserting, deleting and modifying a single letter. Therefore the measure is called *edit* or *Levenshtein distance* (Fig. 1b).

Edit distance penalizes each of the editing operations with the same cost. *Weighted edit distance* is an extension to the edit distance which distinguishes editing operations by assigning different (but constant) costs to each of them.

For many applications weighted edit distance is sufficient, not so for protein sequence alignment where we need to incorporate a *weighting system* which assigns a specific cost to each pair of amino-acids which is used when a pair of amino-acids is aligned. The need arises from the fact that protein sequences comprise a kind of semantics which stems from the evolutionary history of protein sequences. Knowing evolution history of many sequences allows us statistically derive probability that an amino-acid will mutate into another one in a given time period. Based on this probability, we are able to assign score to each pair of amino-acids. In bioinformatics, we present this fact by $20x20$ *substitution matrix*. There are many different sets of matrices[6] nowadays and the most wide spread ones are PAM [3] and BLOSUM [7].

Final modification, specific to protein sequences alignment, is different costs for opening and extending a gap. This means that in the resulting alignment a position where a gap starts is penalized (usually noticeably) more then the consequent gap positions.

Moreover to this scoring systems, we differentiate between *local* and *global* alignment. The standard conception of aligning sequences is called global alignment. But if we are interested in finding substrings of pairs of sequences that express high level of similarity, we use local alignment. Thus the best local alignment is an alignment of such a pair of substrings that has the highest global alignment score among all possible pairs of substrings.

2.1 Dynamic Programming

There are quadratic complexity algorithms based on dynamic programming for computing both global and local alignments.

[6] Depending whether we are searching for evolutionary close or distant homologues we use different matrices of these sets.

Global Alignment. Dynamic programming algorithm for global alignment of two sequences was published in 1970 by Needleman and Wunsch [12]. It is based on dynamic programing matrix G of size $(m+1) \times (n+1)$ where m and n are lengths of the sequences to be aligned. $G[i,j]$ stores score of the optimal alignment of prefixes of respective lengths (zero-th row and column are designated for initialization). The recursive formula for computing cells of the matrix is:

$$G[i,j] = max \begin{cases} G[i-1,j] + \sigma \\ G[i,j-1] + \sigma \\ G[i-1,j-1] + S[a_i, b_j] \end{cases} \quad (1)$$

In this formula σ stays for gap cost, $i \in \{1..m\}, j \in \{1..n\}$, a and b are sequences to be aligned and S is a substitution matrix. Ergo, at position $[i,j]$ we can align i-th and j-th letter or add a gap to one of the sequences. In the initialization phase, we fill the border cells with $G[i,0] = i * \sigma$, $G[0,j] = j * \sigma$, because alignment of a string of zero length with a string of length i demands inserting i spaces into the empty string.

From the fact that $G[i,j]$ contains score of the resulting optimal alignment of respective prefixes follows that score of the optimal global alignment can be fetched from $G[m,n]$. In this paper, the score is what matters to us but if we were interested in the alignment itself, we could backtrack the matrix in the same way it was filled according to the formula 1 (first line means adding space into 'vertical' sequence, second adding space into 'horizontal' sequence and third means aligning i-th and j-th letters of the sequences).

The original formula 1 lacks distinction for different costs for opening and extending a gap. In order to express it, we must incorporate two matrices for vertical and horizontal gaps [5]. Role of these matrices is to decide whether it makes sense to start a new gap, or it would be better to continue an already started gap[7]:

$$H[i,j] = max \begin{cases} G[i,j-1] + \sigma \\ H[i,j-1] + \delta \end{cases} \quad (2) \qquad V[i,j] = max \begin{cases} G[i-1,j] + \sigma \\ V[i-1,j] + \delta \end{cases}, \quad (3)$$

where δ stays for the cost of continuing a gap. And finally, we also need to change the recursion for G to incorporate the H and V matrices:

$$G[i,j] = max \begin{cases} V[i,j] \\ H[i,j] \\ G[i-1,j-1] + S[a_i, b_j] \end{cases}. \quad (4)$$

[7] We use matrices here but (as follows from the recursion) one-dimensional arrays would be sufficient if we would align the sequences line by line or column by column (for that matter, this is also true for the G matrix where two one-dimensional array would be sufficient).

Local Alignment. As mentioned earlier, in bioinformatics we are usually more interested in searching common or highly conserved sections of protein sequences. For this purpose, global alignment is not optimal since relatively short but highly similar parts can be 'lost' in order to properly align rest of the sequence. In 1981, Smith and Waterman [16] published dynamic programming algorithm solving local alignment problem. This algorithm is actually a slight modification of Needleman and Wunsch. If at any position score of the global alignment algorithm is positive, the optimal local alignment will certainly not start at that position because the already aligned parts score above zero. But if at any position the score should be negative, it makes sense to start a new local alignment from that position because the final score will be higher by the absolute value of the particular score. Let's add just stated decision into the recursive equation:

$$L[i,j] = max \begin{cases} V[i,j] \\ H[i,j] \\ L[i-1,j-1] + S[a_i, b_j] \\ 0 \end{cases} .^{8} \tag{5}$$

To get the best local alignment we also need to stop aligning at the position with the highest score. Therefore, optimal local alignment score is not at position $L[m,n]$ but at position $[i_{max}, j_{max}]$ with the highest $L[i_{max}, j_{max}]$ value. If we start backtracking from the position $[i_{max}, j_{max}]$ to position $[i_0, j_0]$ where $L[i_0, j_0] = 0$, we get route of the optimal local alignment.

3 Speed-Up by Using Common Parts

As mentioned earlier, there exist areas where rigorous alignment by Smith-Waterman algorithm is needed. Here, we can not sake accuracy for speed and thus the remaining ways how to speed up the search are indexing, parallelism, or speeding up the distance computation itself.

Following the knowledge of distances among objects which are precomputed and stored, the indexing methods filter out as many objects as possible without even fetching them from the database. Methods with primary target to decrease number of distance computations are called metric access methods (MAM) because of incorporating axioms of metric to filter out groups of objects. Nevertheless, MAM's are not applicable to protein alignment problem since it is not metric at all, and it is difficult to turn it into metric, as has been show in [8]. There have also been attempts to turn the distance into metric [18] by modifying the substitution matrix which is based on observation done in [15]. However, this method is applicable only to global alignment and just to q-grams (not the whole sequences). Other approaches have similar drawbacks and up to date there is no indexing method that could replace BLAST (as far as we know).

There are two possible approaches how to parallelize the database search with Smith-Waterman. First one is to parallelize the whole database search by running

[8] This method is applied to initialization phase too, so the border cells are filled with zero values.

multiple alignments at the same time on more than one computational unit. The speed-up is directly proportional to the number of the computational units involved (example of this approach is MPSrch [11][9]). Second approach demands hardware modifications and focuses on speeding up single distance computation (computing the dynamic programming matrix). Hardware platform enabling this solution is FPGA (Field Programmable Gate Array) [13] [4] or standard CPU enabling certain degree of parallelism [14].

As far as we know, there have not been many attempts to improve the alignment of protein sequences itself without a specialized hardware. However, interesting improvement was achieved in [9].

3.1 Basic Algorithm

Method presented in this paper saves computation costs as indexing methods do, but instead of omitting distance computations it reuses parts of the distance matrix computed by Smith and Waterman. This decreases number of operations needed for virtually every alignment done during the whole database search.

So, the idea is to store parts of the matrix and use it later. But almost every submatrix in the dynamic programming matrix is context dependent - its content is dependent not only on the amino-acids to be aligned, but also on the calculation made so far. This makes it impossible to store any inner submatrix for later use unless content of the border cells are the same in a future alignment. The only set of submatrices having the same left and top context are submatrices starting at position [0,0]. While the query sequence stays the same in the search, if two sequences share a common prefix, their Smith-Waterman matrix will be identical up to the point where they start to differ. Let's imagine the query sequence to be at the top side of the matrix and the database sequence at the left side then if two sequences s_1, s_2 share common prefix of length n then the main idea is as follows:

1. Align s_1 with the query sequence (fill the dynamic programming matrix).
2. Replace s_1 in the matrix by s_2.
3. Start Smith-Waterman with s_2 from the $(n+1)$-th row.

Of course, we need to repeat the same technique with the H and V matrices to make the whole thing work[10].

Notice that we do not need to change the existing algorithm at all and we even do not need a persistent storage for the submatrices corresponding to the common prefixes. All we need is to sort the database according to the prefixes. Then we can traverse the database in prefix order, and if two consequent sequences share a common prefix, we save portion of distance computations proportional to the common prefix length. When traversing this way, the method has no additional memory demands at all.

[9] Since MPsrch is a commercial product, its exact algorithm is not known.

[10] If we store just one-dimensional arrays instead of matrices L, H and V, the algorithm still works fine we just need to store the respective one-dimensional arrays (located at the position where sequences start to differ) and use it in the next step.

We define so called *prefix ratio* which is the proportion between overall length of the prefixes (i.e. if we sort the database according to the prefixes, then each sequence contributes to the overall length with length of the shared prefix with the previous sequence) and the length of the database (sum of lengths of individual sequences). The speed-up is then equivalent to the prefix ratio of the database being searched, and is independent on the query sequence.

3.2 Improvement by Using Inversed Sequences

Further speed-up might be achieved if we would find another parts that are common to some set of sequences and are context independent or the context is the same among them. We realized that the score of the optimal alignment is independent on the direction of the alignment, hence if we denote $r(s)$ inversion of sequence s, then score of the optimal alignment of s_1 and s_2 is equivalent to the score of the optimal alignment of $r(s_1)$ and $r(s_2)$:

Theorem 1. *Let's denote an optimal alignment of strings* $\underline{s_1}$ *and* $\underline{s_2}$ *as* $\underline{a(s_1, s_2)}$*, score of the alignment as* $\underline{opt} = \underline{s(a(s_1, s_2))}$*,* $\underline{r(s)}$ *the reverse string of* $\underline{s}$*. Then* $\underline{opt} = \underline{s(a(s_1, s_2))} = \underline{s(a(r(s_1), r(s_2)))}$.

Proof. First, we show that the score of an arbitrary alignment does not depend on the direction in which it is computed. At each position of an alignment, two letters are aligned or there is a space in one of the sequences. In the first case, since aligning of two letters is not context dependent, the direction does not matter. In the latter case, each position inside a longer gap is scored equally independently of the direction. For border positions of the gaps, the first position in the direction of the alignment is scored differently than the last one, but sum of both is again equal independently of the direction.

Hence, score of an alignment is independent of the direction. Let opt_r be the score of the optimal alignment of the reversed sequences. If $opt < opt_r$, than $s(a(s_1, s_2)) = opt < opt_r = s(a(r(s_1)), r(s_2))$ which is in conflict with the proved fact that $s(a(s_1, s_2)) = s(a(r(s_1), r(s_2)))$. The same is true for $opt > opt_r$. Hence, $opt = opt_r$.

We can use the knowledge of the score of the alignment of reversed strings to improve the presented method. For each sequence in the database we can decide whether to align it with the query sequence in the standard way or whether to reverse both, the query and database sequence and do the alignment. If we *appropriately* divide the database into two groups (sequences to be aligned in the standard way and sequences to be aligned reversely) we might increase the prefix ratio and thus speed-up the whole search.

Partitioning of the database can be done in two stages:

1. Divide a given percent of the database into 2 groups randomly, and then add each of the remaining sequences into a group so that the overall prefix ratio increases.
2. Repeat following step n-times - move a random sequence from one group into the other one if it would increase the overall prefix ratio.

3.3 Inexact Search

Method presented in this paper is dependent on the amount of the prefix ratio of the database, so one of the goals is to increase this ratio. With growing size of the database, the prefix ratio should increase because the probability of sequences having common prefixes increases too. By chopping sequences into more parts, we get bigger database with shorter sequences, hence the prefix ratio should grow. But this method brings a serious drawback - if an optimal alignment of an unbroken sequence will be spanned over the point of split then the sequence might not occur in the result set any more[11].

4 Experimental Results

In our experiments, we focused on how various parameters and methods influence prefix ratio (PR). Remember that PR corresponds to percentage of the speed increase according to the full scan. Experiments were performed on subset of UniProt database [17] with restricted lengths of the sequences to 3000 letters which makes 99.9% (5,340,227 sequences) of the whole database.

In all the experiments where subset of database were used, five independent random subsets of given size were generated and experiments where carried out against each of them. The results were averaged in order to avoid random subsets with higher PR than the average.

4.1 Prefix Ratio

For the first experiment we generated subsets of size 1000, 5000, 10000, 15000, 30000, 50000, 80000, 100000, 200000, 50000 and 1000000. These datasets were used to find out how size influences PR. We expected that when dealing with bigger number of sequences, the probability of the sequences having more common prefixes increases. In Tab. 1, we can see that this assumption is correct, e.g. subset of size 1,000,000 has PR 9.1 whilst subset of size 1000 only 0.9. To receive the highest achievable prefix ratio (according to todays database size), we carried out the same test also against the whole UniProt which gave us 18% speed-up. This speed-up should grow (not unlimitedly) with increasing size of the protein databases.

We also performed PR tests against few semantic based subsets which might lead to higher prefix ratio since sequences from similar organisms might show higher similarity in general. For these test we used subsets of UniProt based on taxonomic divisions[12]. Results, supporting the assumption of higher prefix ratio of semantically closed datasets, can be seen in Tab. 2. Majority of the sets have noticeably higher prefix ratio than random sets of comparable size from Tab. 1 (especially bacteria and viruses datasets).

[11] Sequence appears in the result set if one of it's parts align with the query with score higher than the threshold.

[12] Can be downloaded from EBI FTP.

Table 1. PR (in %) of random subsets

Subset Size	*Prefix ratio*
1000	0.9
5000	1.4
10000	1.7
15000	1.9
30000	2.4
50000	2.8
80000	3.2
100000	3.6
200000	4.6
500000	6.8
1000000	9.1
5340227	18

Table 2. PR (in %) based on taxonomic divisions

Taxonomic Division	*Count*	*Prefix ratio*
archaea	11489	2.4
bacteria	140132	18.7
fungi	19520	2.1
human	17599	1.1
invertebrates	14001	3.8
mammals	16807	8.8
plants	23718	8.7
rodents	22280	4.2
vertebrates	12213	4.4
viruses	11525	9.2

4.2 Reversed Sequences

In order to see how we can improve PR by using reversed sequences as explained in section 3.2, we performed experiments on subsets up to the size of 200,000. First, experiments concerned the initial build of the two groups. In the building stage, a portion of the database is divided into the groups randomly, and the rest is added in the way that would increase the overall PR. Hence we focused on finding the appropriate percentage of the database that should be inserted randomly. In Fig. 2a, we can see that there is no given percentage that would be optimal for all cases. However, in most cases to insert about 50% of the database randomly works just fine. In absolute numbers, after the building stage the PR is worse than the PR with basic algorithm.

In the next experiment, we were shifting sequences randomly between the two groups to improve PR. Results in Fig. 2b clearly show that the ratio increases just

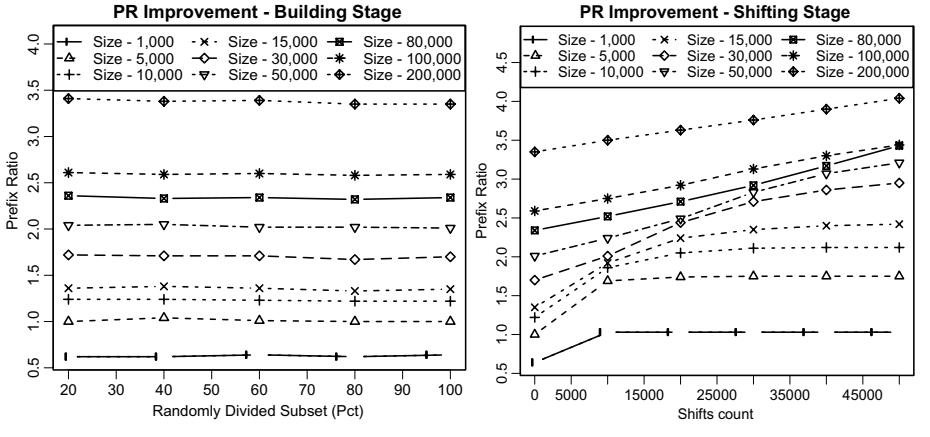


Fig. 2. Prefix ratio - a) after building stage according to different sizes being divided randomly, b) in shifting stage according to increasing number of shifts

Table 3. Prefix ratio growth by using reversed sequences

subset	*prefix ratio*			*subset*	*prefix ratio*		
	build	*shifts*	*no reverse*		*build*	*shifts*	*no reverse*
1000	0.6	1	0.9	50000	2	3.2	2.8
5000	1	1.8	1.4	80000	2.4	3.4	3.2
10000	1.2	2.1	1.7	100000	2.6	4.8	3.6
15000	1.4	2.4	1.9	200000	3.4	6.2	4.6
30000	1.7	3	2.4				

up to a particular value for smaller datasets. We believe that the absolute possible prefix ratio was not reached at this point, but the distribution reached a local optimum. What lead us to this assumption? We limited size of each group[13] because sequences tend to be cumulated in one of the groups (the bigger one). In the point where the optimum was achieved, sizes of the groups were limitary which might be an indication of a local optimum. Even though, we managed to increase PR with this method about 20% according to the basic method. For convenience, in Tab. 3 you can compare PR after building stage, PR achieved by shifting sequence between groups[14] and finally, PR achieved without using reversed sequences.

4.3 Splitting

Last experiments investigated impact of splitting on PR and accuracy. In all the experiments, 5,000,000 alignments were performed and the results were averaged. Tab. 4[15] presents first part of the experiment - for different E-values[16] shows number of sequences that pass the cutoff score, average lengths of these sequences, their average SW score, average score that they need to pass, and number of cells in the dynamic programming matrix that reach the maximal value (hence number of possible optimal alignments). Next rows in the table show

Table 4. Alignment statistics

	E-value											
	6		*8*		*10*		*12*		*14*		*max*	
result size	7252		7455		7607		7685		7771		7906	
∅ length	171		171		172		173		174		25	
∅ SW score	384		389		396		402		411		31	
∅ cutoff score	77.5		76.5		75.6		75		74.2		0	
∅ max values	1.3		1.3		1.3		1.3		1.3		1.5	
	abs	pct	abs	pct	abs	pct	abs	pct	abs	pct	abs	pct
1 splitting	5887	81.2	6001	80.5	6086	80	6173	79.4	6245	79	1159830	23.2
2 splittings	6694	92.3	6860	92	6984	91.8	7112	91.5	7216	91.3	2072025	41.4
3 splittings	6993	96.4	7172	96.2	7305	96	7450	95.9	7570	95.8	2700590	54
4 splittings	7092	97.7	7278	97.6	7420	97.5	7572	97.4	7695	97.3	3204691	64

[13] The size of group A can be at most 2/3 of size of B.

[14] These numbers slightly differ from numbers in Fig 2b because here we performed more shifts (to reach the local optimum).

[15] Experiments in Tab. 4 and Tab. 5 were performed against the whole UniProt database.

[16] In practice usually E-value 10 is used.

Table 5. Splitting accuracy loss

	E-value				
	6	*8*	*10*	*12*	*14*
1 splitting	14.3%	16.2%	15.8%	17.2%	17.2%
2 splittings	13.4%	15.1%	15.8%	15.6%	15.9%
3 splittings	17.1%	17.2%	17%	18%	18.2%
4 splittings	15.9%	16.9%	17.1%	17.6%	18.1%

number of sequences that pass the cutoff score and (at the same time) span over given number of splits[17]. Inspecting number of sequences spanning over a split might lead us to a conviction that also real inaccuracy will show similarly poor numbers (inaccuracy about 90%) which was not confirmed, as Tab. 5 certifies. The accuracy loss (number of sequences that would normally score above a given threshold but do not because of the split) is about 17%. SW score and cutoff score from Tab. 4, together with number of maximal values in one alignment, are responsible for this contradiction. When an optimal alignment is split, it has such a high score that even its parts may score above the cutoff score alone. Moreover, in some cases there are more than one optimal alignment and thus one of them might not be split at all.

Interesting observation is that the accuracy is higher for three splits than it is for two. Such an observation has already been made and taken into account in computing the E-value - probability of an alignment in the middle of a sequence is higher than on its edges.

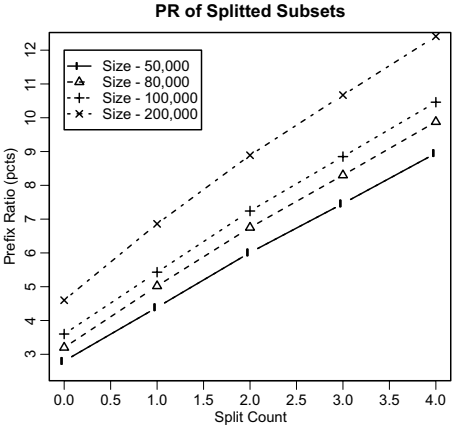


Fig. 3. Relation of number of splits and prefix ratio

Finally, we investigated how size of the database influences PR when splitting. In Fig. 3, we can see that with increasing size of the database also the PR gain increases.

5 Conclusion

We implemented and tested a modification of Smith-Waterman algorithm for large datasets which benefits from shared prefixes and suffixes of the sequences. This modification can be incorporated into existing implementations, thus increasing speed for the price of just slight modifications. If used with methods aligning parallely more sequences in one moment, the speed-up might be lower since neighboring sequences (in the sense of prefix order) might be evaluated

[17] Splitting occurs equally along the sequence.

concurrently. Nevertheless, the speed-up might be up to 20% without any accuracy loss. When inacuraccy included, the speed-up increases approximately two times.

Further speed-up might be achieved by incorporating a kind of lookahead when categorizing sequences into the two groups. This might help to (partially) avoid the observed stagnation in a local optimum.

Acknowledgments. This research has been supported by grant GAUK 57907 provided by the Grant Agency of Charles University.

References

1. Altschul, S.F., Madden, T.L., Schaffer, A.A., Zhang, J., Zhang, Z., Miller, W., Lipman, D.J.: Gapped BLAST and PSI–BLAST: a new generation of protein database search programs. Nucleic Acids Res. 25, 3389–3402 (1997)
2. Barton, G.J.: An efficient algorithm to locate all locally optimal alignments between two sequences allowing for gaps. Computer Applications in the Biosciences 9(6), 729–734 (1993)
3. Dayhoff, M.O., Schwartz, R.M., Orcutt, B.C.: A model for evolutionary change in proteins. Atlas of Protein Sequence and Structure 5, 345–352 (1978)
4. Dydel, S., Bała, P.: Large scale protein sequence alignment using fpga reprogrammable logic devices. In: Becker, J., Platzner, M., Vernalde, S. (eds.) FPL 2004. LNCS, vol. 3203, pp. 23–32. Springer, Heidelberg (2004)
5. Gotoh, O.: An improved algorithm for matching biological sequences. J. Mol. Biol. 162(3), 705–708 (1982)
6. Green, `http://www.genome.washington.edu/UWGC/analysistools/Swat.cfm`
7. Henikoff, S., Henikoff, J.: Amino acid substitution matrices from protein blocks. Proc. Natl Acad. Sci. USA. 89, 10915–10919 (1992)
8. Hoksza, D., Skopal, T.: Index-based approach to similarity search in protein and nucleotide databases. In: DATESO, pp. 67–80 (2007)
9. Itoh, M., Goto, S., Akutsu, T., Kanehisa, M.: Fast and accurate database homology search using upper bounds of local alignment scores. Bioinformatics 21(7), 912–921 (2005)
10. Lipman, D.J., Pearson, W.R.: Rapid and Sensitive Protein Similarity Searches. Science 227, 1435–1441 (1985)
11. MPSrch, `http://www.ebi.ac.uk/MPsrch/`
12. Needleman, S.B., Wunsch, C.D.: A general method applicable to the search for similarities in the amino acid sequence of two proteins. Journal of Molecular Biology 48(3), 443–453 (1970)
13. Ramdas, T., Egan, G.: A survey of fpgas for acceleration of high performance computing and their application to computational molecular biology. In: TENCON 2005 IEEE Region, vol. 10, pp. 1–6 (2005)
14. Rognes, T., Seeberg, E.: Six-fold speed-up of smith-waterman sequence database searches using parallel processing on common microprocessors. Bioinformatics 16(8), 699–706 (2000)
15. Sellers, P.H.: The theory and computation of evolutionary distances: Pattern recognition. J. Algorithms 1(4), 359–373 (1980)

16. Smith, T.F., Waterman, M.S.: Identification of common molecular subsequences. J. Mol. Biol. 147(1), 195–197 (1981)
17. Wu, C., Apweiler, R., Bairoch, A., Natale, D.A., Barker, W.C., Boeckmann, B., Ferro, S., Gasteiger, E., Huang, H., Lopez, R., Magrane, M., Martin, M.J., Mazumder, R., O'Donovan, C., Redaschi, N., Suzek, B.: The universal protein resource (uniprot): an expanding universe of protein information. Nucleic Acids Res. 34(Database issue)(1), D187–D191 (2006)
18. Xu, W., Miranker, D.P.: A metric model of amino acid substitution. Bioinformatics 20(8), 1214–1221 (2004)

Invited Keynote Talk: Computing P-Values for Peptide Identifications in Mass Spectrometry

Nikita Arnold[1,2], Tema Fridman[1], Robert M. Day[1], and Andrey A. Gorin[1]

[1] Computational Biology Institute, Computer Science and Mathematics Division, Oak Ridge National Laboratory, Oak Ridge, TN, 37830
[2] Soft Matter Physics/Experimental Physics, J. Kepler University, Altenbergerstraße 69, A-4040, Linz, Austria

Abstract. Mass-spectrometry (MS) is a powerful experimental technology for "sequencing" proteins in complex biological mixtures. Computational methods are essential for the interpretation of MS data, and a number of theoretical questions remain unresolved due to intrinsic complexity of the related algorithms. Here we design an analytical approach to estimate the confidence values of peptide identification in so-called database search methods. The approach explores properties of mass tags — sequences of mass values (m_1 m_2 ... m_n), where individual mass values are distances between spectral lines. We define p-function — the probability of finding a random match between any given tag and a protein database — and verify the concept with extensive tag search experiments. We then discuss p-function properties, its applications for finding highly reliable matches in MS experiments, and a possibility to analytically evaluate properties of SEQUEST X-correlation function.

Keywords: mass-spectrometry, database search, confidence values.

1 Introduction

Mass-spectrometry based proteomics is the driving engine behind an increasingly rich variety of biological experiments: from a pull-down "hunt" of the protein complexes to whole cell protein expression profiles. The resulting information flow, while disparate in nature and usually huge in volume, often has a common structure of the underlying raw data — individual spectra of short peptides converted into sequences assigned to them by various algorithms.

In a typical experiment, cellular proteins are cut into relatively short peptides (10-20 amino acids), and each analyzed peptide results in an MS spectrum as presented in Fig 1. Peaks are footprints of smaller chemical fragments, where peak position reflects each fragment's mass-to-charge ratio that can be converted to a mass value. Ions corresponding to the breaking of a peptide bond (two highlighted peaks on the picture) are called b-ions and their complements to the full peptide are called y-ions. Typically these two types of ions have relatively high intensity as peptides break more

I. Măndoiu, R. Sunderraman, and A. Zelikovsky (Eds.): ISBRA 2008, LNBI 4983, pp. 100–109, 2008.

easily across peptide bonds. All identification methods utilize this property in some way, but other types of ions also saturate a spectrum (outnumbering "noble" b- and y-ions by a ratio of 20:1), and some of them can be very strong as well.

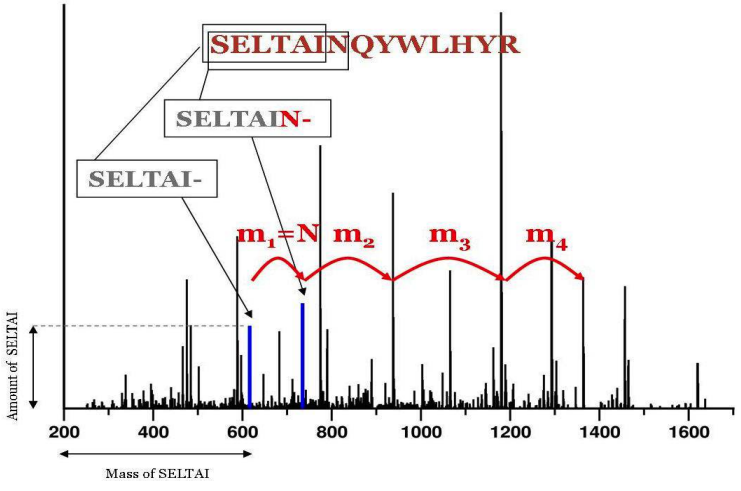


Fig. 1. MS experimental data. Highlighted peaks are formed by 2 b-ions; the distance between them is the mass of N residue at the right terminus of the partial peptide SELTAIN.

Peptide identification aims to infer peptide amino acid sequence from its spectrum. Database search methods [1-5] dominate the field, with an overwhelming majority of experiments using one of them. In database search methods, a peptide is assumed to belong to a known protein database (DB). The SEQUEST program (developed in John Yates group, [2]) uses the following algorithms (some details are simplified):

(1) The experimental spectrum is re-calibrated, so several of the strongest peaks are given an intensity of 1, and other peaks are rescaled accordingly.
(2) The program forms a large list of candidate peptides selected from the target protein DB.
(3) Theoretical spectra are generated for each of the candidate peptides. They usually include only b- and y-ions, and only ion positions are important, as currently there is no reliable way to model relative peak intensity.
(4) Theoretical constructs are matched against the experimental spectrum to compute a matching score. The X-corr reflects the total intensity of experimental peaks that were matched within experimental precision of theoretical positions. The candidate peptide with the highest X-corr value is selected as the output solution.

A typical proteomics experiment incorporates millions of individual peptide identifications, and the reliability of individual assignments is crucially important. We have described the SEQUEST algorithm, because (1) it is hugely popular (probably ~ 50% of the market); and (2) many other search methods were inspired by SEQUEST and

work in a similar fashion. The description is also instructive in regards to algorithm complexity and challenges that one needs to overcome to estimate the reliability of the answers. Every spectrum will be assigned some candidate peptide, but what cutoff of the X-corr values would guarantee, for example, that 95% of the assignments are correct?

The standard way of addressing this problem is by introducing an artificial negative control into the experiment [6,7,8]. The identification procedure is run against a database with two parts: a "true" DB of all protein sequences, which actually were present in the source sample, and "false" one, containing negative controls (also called decoy DB). The decoy database contains proteins that cannot be possibly matched by the sample in question. Several research groups extensively investigated the best approaches to create negative control DB and use them for learning reliable values of the X-corr [9,10,11,12,13,14]. The matches to the decoy part are incorrect by design of the experiment, and X-corr cutoffs can be mapped to the sensitivity values by assuming, for example, that the total number of false matches was twice as high as observed in the false DB.

However, this approach has numerous drawbacks. The X-corr values recorded for a given spectrum depend on many factors: size of the database (in a non-trivial way that is hard to figure out), the particular type of MS device, the type of precursor ion, contaminations (MS experiments are ultra sensitive), and even on the organism, that was the source of the tested sample. On an intuitive level it is clear that the X-corr cutoff should be determined by parameters of a particular spectrum. But this road has insurmountable difficulties for an empirical approach, as there is no obvious way to divide spectra into classes.

Strictly speaking the mapping of the X-corr to the probability has to be done for each modification of the experimental system, but it is not an easy demand. The practical approach is to take a "high enough" X-corr cutoff and hope that the fraction of correct matches will not fall too low. Usually only a small fraction of spectra passes the required cutoff (10-20%), and dissatisfaction goes both ways: it is often a rather small "crop", and it is still not obvious how reliable the obtained matches are.

We propose to explore a different approach to the problem by examining database matches of somewhat simpler objects, which we call mass tags. The idea of "tags" was pioneered by Mann's group [15] and further developed in [16]. We derive an analytical expression for the probability of tag random match and explore the properties of the corresponding function. We also propose a database search model that gives an analytical estimate for the fraction of correct matches and outline how SEQUEST X-corr function can be evaluated in the same mathematical framework.

2 Probability Function of Peptide Mass

We define mass tag as a sequence of mass values (m_1 m_2 ... m_l), where individual values (called connector masses or simply connectors) are distances between spectral lines in a specific peak subset. One such subset and the corresponding mass tag of length 4 are illustrated for the spectrum in Fig. 1. We define a match between tag and database in a different way than match between spectrum and database is usually

defined. A DB entry *seq* matches a tag (m_1 m_2 … m_n) if it contains *n* consecutive protein sequences (seq_1 seq_2 … seq_n), where each sequence seq_k has a mass m_k within experimental precision of the MS device (for our purposes we assume it to be 0.5 Da).

2.1 *P*-Function for a Single Connector

To explore properties of tag matches we will introduce another definition, which is central for all subsequent developments. We define peptide probability mass function — *p*-function — as probability to observe a peptide in the window (*m-dm, m+dm*) starting at an arbitrary point of the protein database. This probability is a function of both *m* (mass of the desired peptide) and *dm* (detection precision), but it does not depend on the size of the database. For example, if *m* is the mass of amino acid Ala and *dm*=0.5 Da, then the probability is equal to Ala frequency in the database.

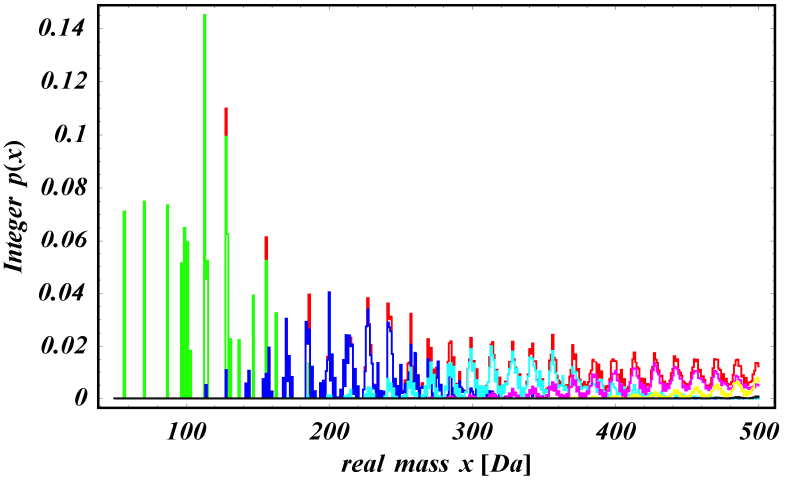


Fig. 2. The contribution of different steps into p-function distribution: green curve - single, blue –double, cyan - triple, magenta - 4 residue steps, and yellow - 5. The overall value of the theoretical *p*-function is shown in red.

The *p*-function dependence from *dm* is an interesting topic that deserves a separate discussion. For the rest of the paper we assume *dm*=0.5 Da. This value is a good choice for two reasons: (1) 0.5 Da is the precision of the most common mass-spectrometers used in proteomics research; (2) peptide masses are naturally concentrated to the centers of so-called Mann bins [17,18,19], which are separated by ~1Da distance on the mass axis. In this sense masses of all peptides, derived ions, and distances between ions are nearly integer (Mann's bin mass is ~1.0005 Da). Tags have connectors that could be expressed as an integer number of Mann bins, and *p*-function can be computed for *m* values centered on such bins with the *dm*=0.5. However it is worth to note that our methodology will straightforwardly accommodate any value of *dm*.

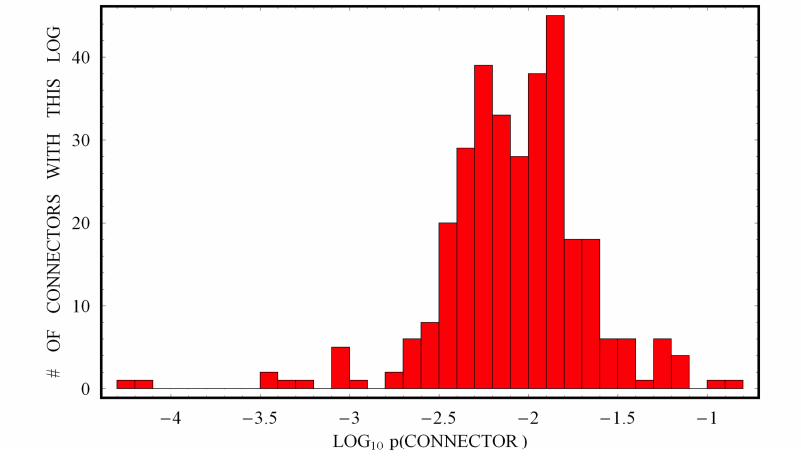


Fig. 3. Distribution of *p*-function values for 500 mass connectors on $\log_{10}$ scale. The Y-axis bin size was 0.1 with bin centers at 0.05, -0.05, -0.15, etc. The distribution contains data for 321 connectors with non-zero *p*-function values.

Remarkably the values of *p*-function are almost independent of all protein DB properties, except frequencies of 20 standard amino acids. We have demonstrated this independence by computing *p*-function recursively without keeping track of particular amino acid combinations leading to a given mass to avoid combinatorial explosion. The process is known as a renewal process [20], as was also pointed out in [16]. The calculations start from bins filled with single amino acids and continue until bin number of 2000.

One can also compute an "experimental" *p*-function for a given database. One just has to generate all peptides, compute their masses, and figure out occupancy frequencies for all [*m-dm, m+dm*] windows of interest. For example, for human genome DB the *p*-function can be obtained as a histogram of approximately ~2×10^8 values (approximate number of peptides in range mass 0 to 2000), which are distributed over 2000 bins. After normalizing histogram by the DB length, we obtained p-function that was almost indistinguishable from the theoretically computed (shown in Fig. 2). Some of the bins are empty. They correspond to mass connectors that could never be observed for a true protein tag. There are many such bins at masses below 200, as this region is occupied mostly by short amino acids combinations.

It is instructive to understand why the model that assumes total independence of the consecutive amino acids provides such a good approximation to reality, while it is known that the real protein text has short and long range sequence correlations. The reason is a "combinatorial elimination" of the correlation artifacts. For example, though combinations like AAAA, QQQQ, and similar ones are much more frequent than it would be expected from uncorrelated model, their contribution changes the *p*-function values only a little, because there are much more other combinations in the same mass bin, which do not show any statistical bias.

The distribution of $\log_{10}$ p values for all 321 non-empty bins found in the interval of m values between 0 and 500 is presented in Fig 3.

2.2 Probability of a Random Database Match

Calculated *p* functions allow us to compute the filtering power of an arbitrary tag (x_1 x_2 ... x_l), where l is the length of the tag. As we assume that there are no correlations between adjacent connectors (highly reasonable assumption for almost all connectors), the total probability of a random realization for a general tag is:

$$p_{\Pi} = \prod_{i=1}^{l} p(x_i)$$

When the tag *p*-function is calculated, it is straightforward to estimate the probability of a random match for a given database. As the match can start from any of N_D amino acids in the database, the average expected number of matches is $r = N_D p_{\Pi}$. The distribution over the observed number of *m* matches is binomial with the probability p_{Π} and the number of Bernoulli trials N_D (size of the DB), but since p_{Π} is small, and N_D is large, it is possible use its Poisson approximation:

$$p(m) = \frac{r^m}{m!} e^{-r}$$

The probability to observe at least one random match is:

$$p_{m>0} = 1 - p(0) = 1 - e^{-r} \approx N_D p_{\Pi}$$

The last expression is an approximation for the case of strong filtering $N_D p_{\Pi} << 1$. The same result can be obtained from the Binomial distribution. $p_{m>0}$ or a complementary quantity $p(0) = e^{-r} \approx 1 - N_D p_{\Pi}$ is a natural measure of database match significance. For a single tag, the match can be considered as non-random with the confidence level $p(0)$.

We have tested our theory by conducting tag match experiments for 10^4 mass tags that were generated in the following way:

(1) Random tag length *l* is chosen from values 2,3,4 and 5.
(2) Random integer numbers are chosen from the interval 57 and 2000, and converted to real values by random selection from centers of the corresponding Mann bins with Gaussian accuracy σ=0.15 (corresponds to the observations on 0.5 Da accuracy devices). These numbers form the testing tag.
(3) Tag probability p_{Π} is calculated. If p_{Π}=0, the tag is discarded, and we return to the step 2.

The results of the tag matches against a large DB (N_D~12×10^6) are presented on Fig. 4. Here, the p-function was constructed from the experimental statistics, but the theoretical function yields similar results. Both the expected and the observed number of matches for the tag of given length span more than 2 orders of magnitude (red, green dots), but, nevertheless, there is a very high correlation between the two.

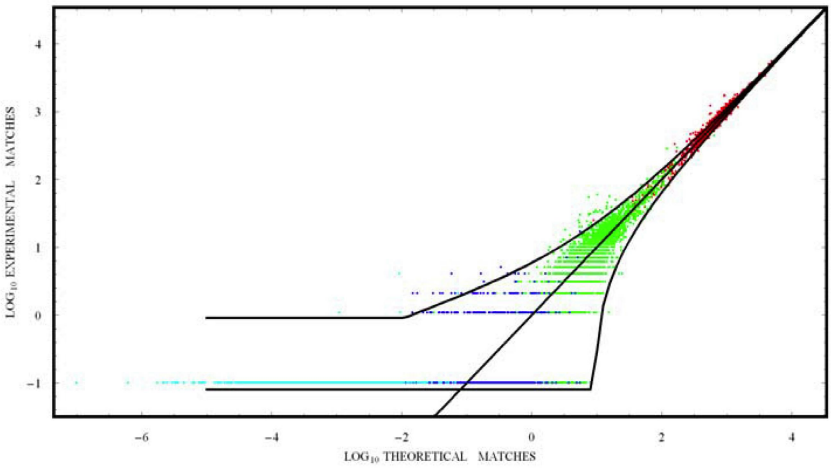


Fig. 4. Log-Log plot of experimental vs. theoretical number of matches. Human proteome DB (N_D=12×10^6). Tag's lengths are shown in color: red (2), green (3), blue (4), cyan (5). Solid lines show the diagonal and the confidence ranges.

Nevertheless, the figure reveals appreciable number of overmatched tags. The confidence ranges at both ends (black curves) are at 10^{-4} level, while the total number of tags was 10^4, so that all points beyond this range are statistically significant. An appreciable amount of such tags is of length 3 (green) or 4 and even 5 (blue, cyan). In some typical cases, the tag of length 3 with the expected number of matches of about 6 demonstrates about 30 of them, which is of course highly improbable. Such cases were investigated and all of them appeared to match identical peptides, resulting from homologies present in any real database.

Fig. 4 contains several unexpected lessons for mass spectrometry identification methods. For green points the number of expected and observed matches concentrates around log-value 1 (ten matches). The green points correspond to tags of 3 mass connectors or, in other words, containing just 2 real ions; and DB size here is larger than in a typical MS experiment. It means that in a real experiments it may be sufficient (in many cases) to find just two true ions to uniquely identify underlying peptide.

2.3 P-Value for High throughput Identification

The calculated probability of random tag matches provides an immediate opportunity to compute the "p-value" of peptide identification in high throughput tag search. Let us consider an asset of tags T={t_i} generated by some tag-selecting algorithm. For each tag we will consider 3 outcomes:

(1) No match: there is no place in the database, where all imposed constraints are satisfied simultaneously

(2) Correct match: tag has matched the correct peptide. By definition it is a single match, and it always happens if the database is complete and does not include mistakes (as we assume everywhere in this paper)
(3) Random match(es): one or more matches that do satisfy all connector conditions. This outcome is not mutually exclusive with outcome (2). Generally, it is possible to have a correct match together with several random ones.

Assuming that for all tags $r_i << 1$, we will get a number of random matches for the whole set $N_{random} = \Sigma r_i$. The sum must be computed through all tags, including those that were never matched. Combining this sum with the observed total number of matches M, we can write a formula for the algorithm sensitivity C, i.e. fraction of correct matches over the total number of matches:

$$C = \frac{M - \sum r_i}{M}$$

This estimate will work for *any* tag-generating algorithm and for any database. It does not require machine learning procedures or an adaptation to a particular database. It also accounts for the precision of ion detections, as well as other possible constraints on peptides (such as tryptic or nontryptic parent peptide). The only requirement is a complete separation between the process of tag construction and tag matching. The part responsible for tag generation should not have "backdoor" access to the database and use only information contained in the spectra itself to generate the tag.

Tag based approaches open new algorithmic possibilities for analysis of the proteomics spectra. Our analysis shows that many different strategies can be pursued, but one has to take into account that informational value of tags differ by 4 (!) orders of magnitude, and it is true even inside the group of tags of length 5. It also seems unwise to consider very few tags in the searches, as the number of random matches can be tightly controlled.

2.4 Computing SEQUEST X-Corr Values

Now we can propose a possible way to calculate analytically "black box" of the SEQUEST X-corr function. We aim to estimate the following: for a given spectrum S and a given database DB, which is a decoy database for the spectrum, find probability to obtain an X-corr value above of a given cutoff CT. The calculation can be accomplished by the following algorithm:

(1) Recalibrate spectrum by the usual SEQUEST procedure;
(2) Determine all groups of peaks that have sum of the recalibrated intensities above CT;
(3) Calculate p_i — p-function value for tags formed by each of those groups. Each subgroup forms a single tag. In addition to all real peaks, two "pseudo" peaks positioned at the zero mass and at the parent mass are added.
(4) The sum Σp_i provides a very good estimate that one of those scenario will realize, and the search procedure will detect an X-corr above CT. More precise estimation can be obtained by considering dependencies between overlapping tags.

3 Conclusions

We presented a rigorous mathematical formalism quantifying the probability of random DB matches for arbitrary tags extracted from the tandem MS spectra. It is shown that the tags consisting of ions separated by some hundreds of Da are in many cases more advantageous then tags consisting of shorter connectors. For example 3-ion tags (and in some cases even 2-ion tags) may suffice for an unambiguous identification in the non-redundant human DB. Developed approach allows a reliable quantification of the expected probability. The random match probabilities for the tags of similar length may differ by several orders of magnitude and are log-normal distributed.

The observed number of random DB matches obeys the Poisson distribution with the mean value calculated as the product of probability of realization for the given mass tag and the database size. This holds even for the tags that differ by several orders of magnitude in the random match probability and observed number of matches. The deviations from this law are shown to be almost exclusively due to homologies present even in the curated non-redundant databases.

Possible extensions of suggested approach include generalizations to arbitrary experimental accuracy, sequence correlations, consideration of database errors, as well as theoretical estimates for background values in many scoring functions currently existing in the field (including SEQUEST X-corr function).

Acknowledgments. This work was funded by a Biopilot project from the DOE Office of Advanced Scientific Computing Research. We also wish to thank Max Fridman for help with the manuscript.

References

1. Hirosawa, M., Hoshida, M., Ishikawa, M., Toya, T.: MASCOT: multiple alignment system for protein sequences based on three-way dynamic programming. Comput. Appl. Biosci. 9, 161–167 (1993)
2. Eng, J.K., McCormack, A.L., Yates, J.R.: An approach to correlate tandem mass spectral data of peptides with amino acid sequences in a protein database. Journal of the American Society for Mass Spectrometry 5, 976–989 (1994)
3. Yates III, J.R., Eng, J.K., McCormack, A.L.: Mining genomes: correlating tandem mass spectra of modified and unmodified peptides to sequences in nucleotide databases. Anal. Chem. 67, 3202–3210 (1995)
4. Tabb, D.L., McDonald, W.H., Yates III, J.R.: DTASelect and Contrast: tools for assembling and comparing protein identifications from shotgun proteomics. J. Proteome Res. 1, 21–26 (2002)
5. Perkins, D.N., Pappin, D.J., Creasy, D.M., Cottrell, J.S.: Probability-based protein identification by searching sequence databases using mass spectrometry data. Electrophoresis 20, 3551–3567 (1999)
6. Keller, A., Nesvizhskii, A.I., Kolker, E., Aebersold, R.: Empirical statistical model to estimate the accuracy of peptide identifications made by MS/MS and database search. Anal. Chem. 74, 5383–5392 (2002)
7. Nesvizhskii, A.I., Keller, A., Kolker, E., Aebersold, R.: A statistical model for identifying proteins by tandem mass spectrometry. Anal. Chem. 75, 4646–4658 (2003)

8. Kapp, E.A., Schutz, F., Connolly, L.M., Chakel, J.A., Meza, J.E., Miller, C.A., Fenyo, D., Eng, J.K., Adkins, J.N., Omenn, G.S., Simpson, R.J.: An evaluation, comparison, and accurate benchmarking of several publicly available MS/MS search algorithms: sensitivity and specificity analysis. Proteomics 5, 3475–3490 (2005)
9. Higdon, R., Hogan, J.M., Van Belle, G., Kolker, E.: Randomized sequence databases for tandem mass spectrometry peptide and protein identification. Omics 9, 364–379 (2005)
10. Higdon, R., Hogan, J.M., Kolker, N., van Belle, G., Kolker, E.: Experiment-specific estimation of peptide identification probabilities using a randomized database. Omics 11, 351–365 (2007)
11. Huttlin, E.L., Hegeman, A.D., Harms, A.C., Sussman, M.R.: Prediction of error associated with false-positive rate determination for peptide identification in large-scale proteomics experiments using a combined reverse and forward peptide sequence database strategy. J. Proteome Res. 6, 392–398 (2007)
12. Qian, W.J., Liu, T., Monroe, M.E., Strittmatter, E.F., Jacobs, J.M., Kangas, L.J., Petritis, K., Camp II, D.G., Smith, R.D.: Probability-based evaluation of peptide and protein identifications from tandem mass spectrometry and SEQUEST analysis: the human proteome. J. Proteome Res. 4, 53–62 (2005)
13. Elias, J.E., Gygi, S.P.: Target-decoy search strategy for increased confidence in large-scale protein identifications by mass spectrometry. Nat. Methods 4, 207–214 (2007)
14. Choi, H., Ghosh, D., Nesvizhskii, A.I.: Statistical validation of peptide identifications in large-scale proteomics using the target-decoy database search strategy and flexible mixture modeling. J. Proteome Res. 7, 286–292 (2008)
15. Mann, M., Wilm, M.: Error-tolerant identification of peptides in sequence databases by peptide sequence tags. Anal. Chem. 66, 4390–4399 (1994)
16. Sunyaev, S., Liska, A.J., Golod, A., Shevchenko, A., Shevchenko, A.: MultiTag: multiple error-tolerant sequence tag search for the sequence-similarity identification of proteins by mass spectrometry. Anal. Chem. 75, 1307–1315 (2003)
17. Frahm, J.L., Howard, B.E., Heber, S., Muddiman, D.C.: Accessible proteomics space and its implications for peak capacity for zero-, one- and two-dimensional separations coupled with FT-ICR and TOF mass spectrometry. J. Mass Spectrom 41, 281–288 (2006)
18. Mann, M.: Useful tables of possible and probable peptide masses. In: 43rd ASMS Conference on Mass Spectrometry and Allied Topics, Am. Soc. Mass Spectr., Atlanta (1995)
19. Zubarev, R.A., Hakansson, P., Sundqvist, B.: Accuracy Requirements for Peptide Characterization by Monoisotopic Molecular Mass Measurements. Anal. Chem. 68, 4060–4063 (1996)
20. Kampen, N.G.v.: Stochastic processes in physics and chemistry. North-Holland, Amsterdam, New York (1992)

PFP: A Computational Framework for Phylogenetic Footprinting in Prokaryotic Genomes

Dongsheng Che[1,2], Guojun Li[1], Shane T. Jensen[3], Jun S. Liu[4], and Ying Xu[1]

[1] Computational Systems Biology Laboratory,
Department of Biochemistry and Molecular Biology and Institute of Bioinformatics,
University of Georgia, Athens, GA 30602, USA
[2] Department of Computer Science, University of Georgia, Athens, GA 30602, USA
[3] Department of Statistics, The Wharton School, University of Pennsylvania,
Philadelphia, PA 19104, USA
[4] Department of Statistics, Harvard University, Cambridge, MA 02138, USA

Abstract. Phylogenetic footprinting is a widely used approach for the prediction of transcription factor binding sites (TFBSs) through identification of conserved motifs in the upstream sequences of orthologous genes in eukaryotic genomes. However, this popular strategy may not be directly applicable to prokaryotic genomes, where typically about half of the genes in a genome form multiple-gene transcription units or operons. The promoter sequences for these operons are located in the inter-operonic rather than inter-genic regions, which require prediction of TFBSs at the transcriptional unit instead of individual gene level. We have formulated as a bipartite graph matching problem the identification of conserved operons (including both single-gene and multi-gene operons) whose individual gene members are orthologous between two genomes and present a graph-theoretic solution. By applying this method to *Escherichia coli* K12 and 11 of its phylogeneticly neighboring species, we have predicted 2,478 sets of conserved operons, and discovered potential binding motifs for each of these operons. By comparing the prediction results of our approach and other prediction approaches, we conclude that it is advantageous to use our approach for prediction of *cis* regulatory binding sites in prokaryotes. The prediction software package PFP is available at `http://csbl.bmb.uga.edu/~dongsheng/PFP`.

1 Introduction

Phylogenetic footprinting is a method for identification of *cis* regulatory elements in promoter regions of orthologous genes across species [1]. This strategy attempts to find conserved sequence motifs in the provided promoter regions based on the assumption that functional elements, such as transcription factor binding sites, evolve more slowly than non-functional elements over time. A prerequisite for using a phylogenetic footprinting approach is the mapping of orthologous genes across multiple genomes (often called *reference* genomes).

I. Măndoiu, R. Sunderraman, and A. Zelikovsky (Eds.): ISBRA 2008, LNBI 4983, pp. 110–121, 2008.

A number of orthology mapping approaches, mainly sequence similarity-based such as COG [2] and OrthoMCL [3], have been widely used. By applying such orthology mapping methods to eukaryotic genomes, a number of research groups have carried out studies on identification of *cis* regulatory motifs at a genome scale. For example, Wang *et al.* [4] developed PhyloNet to search for regulatory motifs in *Saccharomyces cerevisiae* by using three other yeast genomes as reference genomes and identified more than 90% of the known TFBSs in *Saccharomyces cerevisiae*. Using several mammalian genomes as references, Xie *et al.* [5] successfully identified a number of transcription regulatory motifs in the human genome.

A similar phylogenetic footprinting strategy may not be directly applicable to prokaryotic genomes due to their different genomic structures from the eukaryotic ones. Typically about half of the genes in a prokaryotic genome are *polycistronic*, *i.e.*, they are organized into multi-gene transcriptional units (or multi-gene operons), genes of each of which share a common promoter and terminator. Multi-gene operons add a new challenge to the identification problem of orthologous promoter regions: promoters are associated with operons rather than individual genes and may not necessarily be conserved across multiple genomes. Thus, relationships between operons across genomes are more complex in general than those between orthologous genes. In addition, the sequence similarity-based approach cannot correctly characterize orthologous relationships in some cases. For prokaryotes, the true orthology can be elucidated by deriving conserved operons across multiple genomes. This is because that homologous genes are more likely to be orthologous if their neighboring genes within an operon are also homologous [6].

Numerous computational methods have been developed to predict operons in prokaryotic genomes, including OFS [7], OPERON [8], OperonDT [9], VIMSS [10], and UNIPOP (manuscript submitted). The prediction accuracy of the best programs has reached 90% on several model genomes such as *E. coli* and *Bacillus subtillis* [11]. It has been previously observed that "conserved" operons may only have their gene list conserved but not necessarily the gene order within the list. In this study, we consider both cases: *category-1* for conserved operons with both conserved gene list and order and *category-2* for conserved operons with only conserved gene list (Figure 1A and 1B). In addition, we also have considered *category-3* for partially conserved operons, which is defined as follows: two operons from different genomes are partially conserved if they have at least one pair of orthologous genes (Figure 1C). Clearly, the multiple scenarios of operon conservation complicate the derivation of orthologous upstream sequences for the purpose of phylogentic footprinting analysis in prokaryotic species.

Previous work on extracting promoter sequences of orthologous genes for phylogenetic footprinting analysis has been done in a simplistic manner. Basically, orthologous genes are collected using sequence similarity-based approaches, then the intergenic sequences of individual genes with the upstream region of its predicted operon are concatenated [12,13,14]. This strategy has also been used in a recent computational tool 'microFootPrinter' [15]. To address the issue of

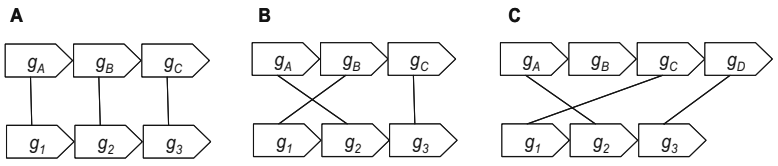


Fig. 1. Three categories of operon conservation. Boxes represent genes and consist of an operon. Lines indicate sequence similarity between two genes. (A) Conserved with both gene list and order; (B) Conserved with gene list only; and (C) Partially conserved.

including upstream sequences for internal genes in an operon, Jensen *et al* [16] considered only the "promoter" regions of genes with upstream intergenic regions longer than 50 bp (called *beginning* genes of an operon). This approach is also problematic since it considers only operons that have both conserved gene list and gene order. There remains a need for more careful and more accurate treatment of the "corresponding" promoters of orthologous genes in prokaryotes.

In this paper, we derive conserved operons among multiple genomes for phylogenetic footprinting analysis and provide a superior treatment of promoter regions of orthologous genes. To fully consider all operons with different levels of evolutionary conservation, we designed an algorithm, *OPERMAP*, to find operons across reference genomes. By applying this algorithm, we have identified 2,478 *E. coli* operons that are conserved across multiple (reference) genomes. In addition, we have developed a pipeline consisting of multiple motif discovery programs for the prediction of conserved sequence motifs. Performance comparison on known binding sites of *E. coli* suggests that our approach tend to generate more reliable orthologous promoter regions (*i.e.*, regions containing the binding sites for orthologous TFs) than previous approaches for motif finding at the genome scale in prokaryotes.

2 Methods

We divide our procedure of phylogenetic footprinting in prokaryotes into five steps:

1. Selecting reference genomes for a target genome;
2. Predicting operons of all selected genomes at genome-scale;
3. Predicting conserved operons across selected genomes;
4. Obtaining promoter sequences of conserved operons;
5. Predicting binding sites using our motif-finding pipeline.

Below, we present the details of each step.

Reference Genome Selection. Selecting suitable reference genomes for comparison to the target genome of interest is a key step in the phylogenetic footprinting process. A candidate reference genome should be phylogenetically close to the target genome. A large list of candidate genomes is not essential since using

a large number of genomes for motif discovery does not seem to improve performance [17]. This has also been observed in our experiments (data not shown). Accordingly, our selection strategy is to choose 10-15 reference genomes belonging to the same class with similar genome sizes to that of the target genome.

In this study, *E. coli* K12 is our target genome and 11 other γ-proteobacteria were chosen as reference genomes. The names and genome sizes of 12 genomes are listed as follows: *Aeromonas hydrophila* ATCC_7966 (4.6 Mb), *Erwinia carotovora atroseptica* SCRI1043 (4.9 Mb), *E. coli* K12 (4.5 Mb), *Photobacterium profundum* SS9 (6.3 Mb), *Photorhabdus luminescens* (5.6 Mb), *Pseudomonas fluorescens* Pf-5 (6.9 Mb), *Salmonella enterica* Choleraesuis (4.9 Mb), *Shewanella ANA* 3 (5.2 Mb), *Shigella sonnei* Ss046 (4.9 Mb), *Sodalis glossinidius morsitans* (4.2 Mb), *Vibrio parahaemolyticus* (5.1 Mb) and *Yersinia pestis* Antiqua (4.8 Mb).

Operon Prediction. For each of the selected genomes, operon prediction at the genome scale is performed using our own program UNIPOP (manuscript submitted). We choose UNIPOP because it outperforms other operon prgrams in terms of prediction accuracy. In addition, unlike most of operon programs, UNIPOP does not need extra feature information (*i.e.*, gene function annotation), which is not available for newly sequenced genomes. The key idea of UNIPOP is to predict operons through identification of conserved gene clusters across multiple genomes. Briefly, given a target genome and N reference genomes, we predict N versions of operon maps for the target genome by comparing and deriving conserved gene clusters between the target genome and each of the reference genomes. We consider two sets of contiguous genes from two genomes to be conserved gene clusters (or operons) if the following conditions are satisfied: a). Each member of a gene cluster is transcribed in the same direction; b). The total intergenic distance within each group is less than the maximum allowed distance (MAD); c). The number of mappings of homologous gene pairs between two groups is at least two. We then obtain a consensus version of operon map using a voting scheme on N versions of operon maps. In this study, operon structures for each of the 12 genomes were predicted by using 348 reference genomes from the NCBI GenBank database (`http://www.ncbi.nlm.nih.gov/genomes/lproks.cgi`).

Identification of Conserved Operons. Having predicted operon structures for the 12 species, we need to identify "orthologous" operons among these prokaryotes. We have developed an algorithm, called *OPERMAP*, to identify the corresponding conserved operon in a particular reference genome for a given query operon in the target genome. We now describe the *OPERMAP* approach in detail as follows.

The input to the algorithm consists of

1. a query operon U in the target genome,
2. a collection of all predicted operons $[V_1, V_2, \ldots, V_k]$ in the reference genome, and
3. a threshold for the degree of conservation (TDC) between two operons.

The output of the program is the operon pair (U, V^*) between the query operon U and the best conserved operon V^* from the reference genome. The algorithm proceeds as follows:

1. Calculate the degree of conservation between query operon U and each candidate operon $[V_1, V_2, \ldots, V_k]$ in the reference genome.
   (a) For each operon $V_i \in [V_1, V_2, \ldots, V_k]$, construct a bipartite graph $G_i = (U, V_i, E_i)$, where all the genes in U and all the genes in the i-th operon V_i are represented as vertices. A pair of genes is considered to be *homologous* if their reciprocal BLAST e-values are both $< 10^{-6}$, and a homologous relationship between a gene in U and a gene in V_i is represented by an edge in E_i. The weight of each edge in E_i is set to be the average of -log(e-value) of the BLAST between the pair of genes.
   (b) Calculate the maximum weighted maximum cardinality bipartite matching (*mwmcm*) M_i on each graph G_i, in a similar fashion to that of [18]. Each matched edge in *mwmcm* reflects the orthology relationship between the pair of genes.
   (c) Calculate the degree of conservation $DC_i = |M_i| \,/\, max\,(|U|, |V_i|)$, where $|X|$ is the cardinality of the set X.
2. The best conserved operon pair (U, V^*) is the operon pair with the highest degree of conservation DC_i. This best operon pair is reported only if the degree of conservation is higher than the predefined threshold TDC; otherwise, no conserved operon pair is returned.

The core of this algorithm is to calculate *mwmcm*. A *matching* in a graph $G = (V, E)$ is a subset M of the edges E such that no edges in M share a common vertex, and a *maximum cardinality matching* (*mcm*) is a matching with the highest possible cardinality. An *mwmcm* is a *mcm* with the maximum total weight (see Figure 2 for an example). In this study, the edge relationship in an *mwmcm* represents the orthology relationship between the two corresponding operons. Using the scheme of *mwmcm* to identify the best conserved operon in *OPERMAP* has several advantages. First, it is guaranteed to find the maximum number of homologous gene relationships between two operons. Second, it can find the true orthogolous gene pair based on sequence similarities in the case where there are several *mcms*, provided that an appropriate weighting scheme is given.

By applying *OPERMAP* on all reference genomes, we can obtain a set of conserved operons for a given query operon in the target genome. For each query operon out of 2,706 predicted operons in *E. coli*, we have applied *OPERMAP* on the 11 reference genomes. In this study, we want to cover not only fully conserved operons (*category*-1 and *category-2*), but also partially conserved operons (*category-3*). Including partial conserved operons has its biological reasoning. Some large operons can break into multiple smaller operons with some part of these smaller operons still maintaining the same regulation mechanism. For instance, a Crp-regulated *xylFGHR* operon in *E. coli* breaks into *xylFGH* and *xylR* in *H. influenzae*, with *xylFGH* maintaining Crp regulation, but *xylR* not

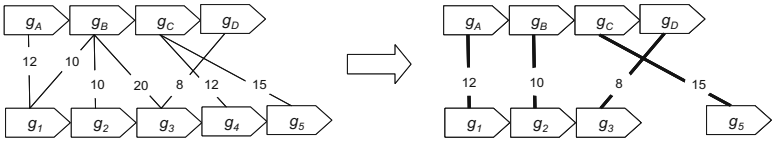


Fig. 2. An illustration of a maximum weight maximum cardinality matching (*mwmcm*). The resulting matching is shown on the right, with the matching size of 4. While the weight of the edge g_B - g_3 is 20, the *mwmcm* does not choose it. Otherwise, the matching size will be 3.

[19]. Setting a low value of *TDC* (*i.e.*, < 0.5) may introduce partial conserved operons with different regulation mechanisms. On the other hand, setting a high value of *TDC*(*i.e.*, > 0.8) will exclude most of partial conserved operons with *category-3* since the sizes of most operons are less than five. We have chosen 0.6 for *TDC* in this study. Experiments on the determination of TDC will not be elaborated in this paper due to space limitation.

Collection of regulatory sequences. The gene annotations and the genomic sequences of the 12 genomes in this study were downloaded from the NCBI GenBank database (`http://www.ncbi.nlm.nih.gov/genomes/lproks.cgi`). For each operon obtained in the previous step, we extract the upstream sequence up to 400 base-pairs (bp) from the translation start site, without overlap of the next upstream gene.

Motif Discovery. The upstream promoter sequences for each conserved operon are the input for our motif discovery pipeline to identify (possibly multiple) TFBSs. The pipeline is similar to our previously developed tool BEST [20], which contains four motif-finding programs: AlignACE [21], BioProspector [22], CONSENSUS [23] and MEME [24], as well as BioOptimizer [25] for optimizing the predictive power of each program. However, BEST is a graphic tool which makes it less suitable for the genome scale motif discovery. Our pipeline overcomes this drawback to produce top-ranked motifs for each sequence dataset in a fully automatic fashion. We outline our motif discovery pipeline in three stages (also see Figure 3).

1. Run the four motif-finding programs mentioned above. Since the motif length in all the four programs must be specified by the user, each program is run multiple times with different motif lengths ranging from 10 to 20 bp. The range of motif lengths chosen is based on the fact that most experimentally verified motifs fall in this range. For each width and each program, the top-ranked motif is collected, giving a set of $4 \times 11 = 44$ top-ranked motifs.
2. The BioOptimizer program is run on each of the 44 motifs. BioOptimizer takes each predicted motif as the starting point and optimizes it using a local hill-climbing technique [25].
3. Rank all 44 optimized motifs based on their score values calculated by BioOptimizer, and output the top five.

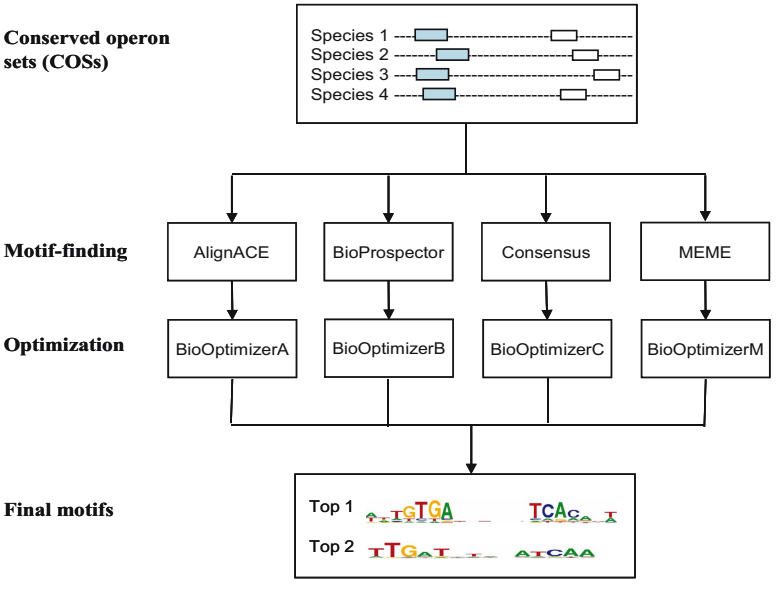


Fig. 3. The workflow of motif discovery. Upstream sequences of conserved operons among closely related species are generated by *OPERMAP*. Datasets are fed into multiple motif-finding programs, and candidate discovered motifs are then optimized by BioOptimizer. The top-ranked motifs based on the score function of BioOptimizer are final identified motifs.

Performance Evaluation. We validate our motif predictions with a similar approach to past motif discovery investigations. We define as *true positives* (TP) the predicted binding sites which overlap with the true binding sites by at least 50%; *false positives* (FP) are the predicted binding sites which have no such overlap; *false negatives* (FN) are the true binding sites that have no overlap with any of the predicted binding sites. We focus on four validation measures, sensitivity (Sn), specificity(Sp), performance coefficient (PC), and F-measuer (F), which are defined as follows:

$$Sn = TP/(TP + FN) \tag{1}$$

$$Sp = TP/(TP + FP) \tag{2}$$

$$PC = TP/(TP + FN + FP) \tag{3}$$

$$F = 2 * Sn * Sp/(Sn + Sp) \tag{4}$$

3 Results

Collection of Conserved Operons. The genome sizes of our 12 genomes range from 4.2 Mb to 6.9 Mb, and the numbers of predicted operons ranged from 1596 to 4468. For each of the 2,706 predicted operons in *E. coli*, we ran

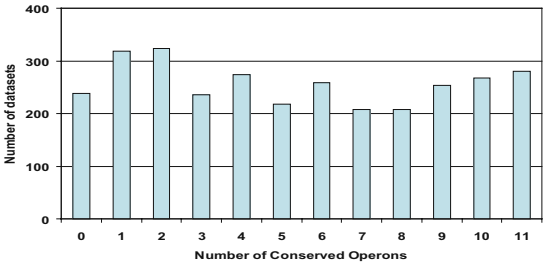


Fig. 4. The operon conservation histogram for 2706 predicted operons of *E. coli*. X-axis indicates the number of conserved operons in 11 other species, and y-axis indicates the number of conserved operons with the conservation number ranging from 0 to 11.

OPERMAP to identify conserved operons in the 11 reference genomes. The distribution of the number of conserved operons across the twelve genomes is shown in Figure 4. Two hundred and thirty-eight operons (8.8%) from *E. coli* do not have a corresponding operon match in any of the 11 reference genomes, which may indicate that those operons are either unique to *E. coli* or have been predicted incorrectly by UNIPOP. At the opposite extreme, 280 operons (10.3%) are conserved across all 11 reference genomes.

Performance of TFBS Predictions. Our evaluation was restricted to predicted motifs in conserved operon sets in *E. coli* since experimentally-verified binding-sites are not available in the 11 reference genomes. We retrieved verified binding sites of *E. coli*, grouped by transcription factors, from the PRODORIC database [26]. We focus on the binding sites regulated by the following ten transcription factors: ArgR, Crp, Fis, Fnr, Fur, IHF, LexA, Lrp, MetJ and SoxS, totally covering 424 verified binding sites. Table 1 shows individual performance statistics for each transcription factor. Prediction accuracies vary among 10 TFs. For example, the prediction sensitivity was 92.6% for LexA, but only 46.7% for Lrp with the known motif. Further studies have shown that Lrp-associated motif was quite degenerate, with the pattern of "NNNNNNTTTATTCT", thus making motif-finding quite difficult. In contrast, LexA-associated motif was a 16-bp palindrome, with a conserved pattern of "CTGTATATATATACAG". In general, our motif discovery pipeline has a high sensitivity but low specificity, similar to other motif prediction results [17]. However, some of this low specificity could be due to unverified but true sites. As more binding sites are verified and deposited in the PRODORIC database, some predicted false positives could become true positives.

Comparison to other approaches. We also compared the performance of our conserved operon-based approach with two orthologous gene-based (specifically sequence similarity-based) approaches, which were used in MicroFootprinter [15] and PHYLOCLUS [16] respectively. In both methods, orthologous genes in other species were identified using a reciprocal BLAST best-hit procedure, with a

Table 1. Prediction accuracy of motif-findings on 10 TFBSs of *E. coli* using the PFP approach

TFs	ArgR	Crp	Fis	Fnr	Fur	IHF	LexA	Lrp	MetJ	SoxS
Sn	0.682	0.64	0.5	0.655	0.761	0.5	0.926	0.467	0.818	0.722
Sp	0.205	0.094	0.113	0.113	0.181	0.066	0.116	0.109	0.138	0.088
PC	0.188	0.089	0.102	0.107	0.172	0.061	0.115	0.097	0.134	0.086
F	0.316	0.163	0.185	0.193	0.293	0.116	0.206	0.177	0.237	0.158

Table 2. Performance comparison between the conserved operon-based (PFP) and the orthologous gene based approaches. The one used in 'Microfootprinter' is named as OrthM, while the one used in 'PHYLOCLUS' is named as OrthB.

Methods	*Sn*	*Sp*	*PC*	*F*
OrthM	0.605	0.109	0.102	0.184
OrthB	0.603	0.105	0.098	0.179
PFP	0.636	0.106	0.100	0.182

Table 3. A list of *glnHPQ* associated orthologous genes and conserved operons predicted by OrthM, OrthB and PFP. *glnH* from *E. coli* was used as a query gene in OrthM and OrthB, while *glnHPQ* from *E. coli* was used as a query operon in PFP. The degree of operon conservation was calculated by *OPERMAP*.

Species	**OrthM**	**OrthB**	**PFP**	**Degree of Conservation**
E. coli	glnH	glnH	glnQ glnP glnH	
A. hydrophila	117619357		117619722 117620284 117620427	0.67
E. carotovora	glnH	glnH	glnH glnP glnQ	1
P. profundum	54302807			
P. luminescens	artI			
P. fluorescens	70728423	70728423	glnH glnP 70733921	1
S. enterica	glnH	glnH	glnQ glnP glnH	1
S. ANA	117919334	117919334	117919332 117919333 117919334	1
S. sonnei	glnH	glnH	glnQ glnP glnH	1
V. parahaemolyticus	28900492			
Y. pestis	108807999	108807999	108807999 108808000 108808001	1

threshold of 10^{-6}. For each method, we generated sequence data sets, ran our motif pipeline for TFBSs prediction, and then evaluated predictions based on 424 binding sites from the PRODORIC database. As shown in Table 2, our approach was more sensitive than the two other ones (63.6% versus 60.5% and 60.3%). The higher sensitivity of our approach over the other two can be attributed to

the reliability of our generated orthologous promoter regions. For example, our approach could detect the true binding-sites of the glutamine permease operon *glnHPQ* in *E. coli*, while the orthologous gene-based couldn't. An investigation of the datasets showed that our approach identified 7 conserved operons for *glnHPQ*, while 'OrthM' identified 10, and 'OrthB' identified 6 "orthologous" genes for *glnH* (shown in Table 3). Further analysis has shown that three 'orthologous' genes (e.g., 117619357, *artI*, 2800492) found by 'orthM' were actually arginine ABC transporters. In addition, both 'orthB' and 'orthM' considered '70728423' from *P. fluorescens* to be an 'orthologous' gene for *glnH*, while our approach did detect a conserved operon *glnHP*-70733921. All these indicate that these four identified genes are not true orthologues, and introduction of the sequences of these genes in OrthB and OrthM lead to the reduction of information content for motif finding.

4 Conclusion

We have presented a computational framework of phylogenetic footprinting in prokaryotes. The major contributions of our work include: a) the introduction of the conserved operon approach, rather than the orthologous gene approach, to collect promoter sequence datasets, and b) the development of motif-discovery pipeline for identifying TFBSs from the sequences we have identified. Performance omparison of TFBSs prediction between our approach and others has shown that our approach could identify more experimentally verified binding-sites.

The better performance of our approach over previous ones is mainly due to the followings: the correct characterization of operon structures in the recent research efforts, and the correct determination of orthology relationships by relying on multiple homologous gene relationships within an operon. In addition, our algorithm *OPERMAP* can nicely incorporate three different categories of conserved operons that maintain the same regulation mechanism.

In our future work, we will predict TFBSs of prokaryotes at the genome scale using our computational framework. By clustering these predicted TFBSs, we can utimately decipher regulons, which is the set of operons whose promoter regions share the similar binding motif patterns regulated by the same transcription factor.

Acknowledgments. This research was supported in part by National Science Foundation (#NSF/DBI-0354771, #NSF/ITR-IIS-0407204, #NSF/DBI-0542119, and #NSF/CCF-0621700) and by a "distinguished scholar" grant from Georgia Cancer Coalition.

References

1. Tagle, D.A., Koop, B.F., Goodman, M., Slightom, J.L., Hess, D.L., Jones, R.T.: Embryonic epsilon and gamma globin genes of a prosimian primate (Galago crassicaudatus). Nucleotide and amino acid sequences, developmental regulation and phylogenetic footprints 203, 439–455 (1988)
2. Tatusov, R.L., Koonin, E.V., Lipman, D.J.: A genomic perspective on protein families. Science 278, 631–637 (1997)

3. Li, L., Stoeckert Jr, C.J., Roos, D.S.: OrthoMCL: identification of ortholog groups for eukaryotic genomes. Genome research 13, 2178–2189 (2003)
4. Wang, T., Stormo, G.D.: Identifying the conserved network of cis-regulatory sites of a eukaryotic genome. In: Proceedings of the National Academy of Sciences of the United States of America, vol. 102, pp. 17400–17405 (2005)
5. Xie, X., Lu, J., Kulbokas, E.J., Golub, T.R., Mootha, V., Lindblad-Toh, K., Lander, E.S., Kellis, M.: Systematic discovery of regulatory motifs in human promoters and 3' UTRs by comparison of several mammals. Nature 434, 338–345 (2005)
6. Wu, H., Mao, F., Olman, V., Xu, Y.: Accurate prediction of orthologous gene groups in microbes. In: Proceedings/ IEEE Computational Systems Bioinformatics Conference, CSB, pp. 73–79 (2005)
7. Westover, B.P., Buhler, J.D., Sonnenburg, J.L., Gordon, J.I.: Operon prediction without a training set. Bioinformatics (Oxford, England) 21, 880–888 (2005)
8. Ermolaeva, M.D., White, O., Salzberg, S.L.: Prediction of operons in microbial genomes. Nucleic acids research 29, 1216–1221 (2001)
9. Che, D., Zhao, J., Cai, L., Xu, Y.: Operon Prediction in Microbial Genomes Using Decision Tree Approach. In: Proceedings of IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology, pp. 135–142 (2007)
10. Price, M.N., Huang, K.H., Alm, E.J., Arkin, A.P.: A novel method for accurate operon predictions in all sequenced prokaryotes. Nucleic acids research 33, 880–892 (2005)
11. Dam, P., Olman, V., Harris, K., Su, Z., Xu, Y.: Operon prediction using both genome-specific and general genomic information. Nucleic acids research 35, 288–298 (2007)
12. McCue, L., Thompson, W., Carmack, C., Ryan, M.P., Liu, J.S., Derbyshire, V., Lawrence, C.E.: Phylogenetic footprinting of transcription factor binding sites in proteobacterial genomes. Nucleic acids research 29, 774–782 (2001)
13. McCue, L.A., Thompson, W., Carmack, C.S., Lawrence, C.E.: Factors influencing the identification of transcription factor binding sites by cross-species comparison. Genome research 12, 1523–1532 (2002)
14. McGuire, A.M., Hughes, J.D., Church, G.M.: Conservation of DNA regulatory motifs and discovery of new motifs in microbial genomes. Genome research 10, 744–757 (2000)
15. Neph, S., Tompa, M.: MicroFootPrinter: a tool for phylogenetic footprinting in prokaryotic genomes. Nucleic acids research 34, 366–368 (2006)
16. Jensen, S.T., Shen, L., Liu, J.S.: Combining phylogenetic motif discovery and motif clustering to predict co-regulated genes. Bioinformatics (Oxford, England) 21, 3832–3839 (2005)
17. Hu, J., Li, B., Kihara, D.: Limitations and potentials of current motif discovery algorithms. Nucleic acids research 33, 4899–4913 (2005)
18. Mehlhorn, K., Näher, S.: Leda: a platform for combinatorial and geometric computing. Cambridge University Press, Cambridge (1999)
19. Tan, K., Moreno-Hagelsieb, G., Collado-Vides, J., Stormo, G.D.: A comparative genomics approach to prediction of new members of regulons. Genome research 11, 566–584 (2001)
20. Che, D., Jensen, S., Cai, L., Liu, J.S.: BEST: binding-site estimation suite of tools. Bioinformatics (Oxford, England) 21, 2909–2911 (2005)
21. Roth, F.P., Hughes, J.D., Estep, P.W., Church, G.M.: Finding DNA regulatory motifs within unaligned noncoding sequences clustered by whole-genome mRNA quantitation. Nature biotechnology 16, 939–945 (1998)

22. Liu, X., Brutlag, D., Liu, J.: BioProspector: discovering conserved DNA motifs in upstream regulatory regions of coexpressed genes. Pac. Symp. Biocomput, 127–138 (2001)
23. Hertz, G.Z., Stormo, G.D.: Identifying DNA and protein patterns with statistically significant alignments of multiple sequences. Bioinformatics (Oxford, England) 15, 563–577 (1999)
24. Bailey, T.L., Elkan, C.: Fitting a mixture model by expectation maximization to discover motifs in biopolymers. In: Proceedings of the Second International Conference on Intelligent Systems for Molecular Biology, pp. 28–36. AAAI Press, Menlo Park, California (1994)
25. Jensen, S.T., Liu, J.S.: BioOptimizer: a Bayesian scoring function approach to motif discovery. Bioinformatics (Oxford, England) 20, 1557–1564 (2004)
26. Munch, R., Hiller, K., Grote, A., Scheer, M., Klein, J., Schobert, M., Jahn, D.: Virtual Footprint and PRODORIC: an integrative framework for regulon prediction in prokaryotes. Bioinformatics (Oxford, England) 21, 4187–4189 (2005)

Accelerating the Neighbor-Joining Algorithm Using the Adaptive Bucket Data Structure

Leonid Zaslavsky and Tatiana A. Tatusova

National Center for Biotechnology Information, National Library of Medicine,
National Institute of Health, Bethesda, MD, 20894, USA
{zaslavsk, tatiana}@ncbi.nlm.nih.gov

Abstract. The complexity of the neighbor joining method is determined by the complexity of the search for an optimal pair ("neighbors to join") performed globally at each iteration. Accelerating the neighbor-joining method requires performing a smarter search for an optimal pair of neighbors, avoiding re-evaluation of all possible pairs of points at each iteration.

We developed an acceleration technique for the neighbor-joining method that significantly decreases complexity for important applications without any change in the neighbor-joining method. This technique utilizes the bucket data structure. The pairs of nodes are arranged in buckets according to values of the goal function $\delta_{ij} = u_i + u_j - d_{ij}$. Buckets are adaptively re-arranged after each neighbor-joining step. While the pairs of nodes in the top bucket are re-evaluated at every iteration, pairs in lower buckets are accessed more rarely, when the algorithm determines that the elements of the bucket need to be re-evaluated based on new values of δ_{ij}. As a result, only a small portion of candidate pairs of nodes is examined at each iteration.

The algorithm is cache efficient, since the bucket data structures are able to exploit locality and adjust to cache properties.

Keywords: neighbor-joining algorithm, bucket data structure, adaptive, cache-efficient.

1 Introduction

The neighbor-joining algorithm [1], [2] is one of the most popular distance methods for the creation of phylogenetic trees. It is a greedy agglomerative algorithm that constructs a tree in steps [3]. The algorithm is based on the minimum-evolution criterion for phylogenetic trees. It is well-tested and studied theoretically, provides good results and is statistically consistent under many models of evolution [4], [5], [6], [7], [8]. Several algorithms have been developed as improvements to the classical neighbor-joining method [9], [10]. Since the neighbor-joining method is much more efficient than other algorithms of comparable quality, it is widely used for phylogenetic analysis as the tool of choice for preliminary analysis, with results being verified and refined by maximum likelihood and Baysian methods [11].

I. Măndoiu, R. Sunderraman, and A. Zelikovsky (Eds.): ISBRA 2008, LNBI 4983, pp. 122–133, 2008.

However, the usage of the neighbor-joining method within interactive exploratory analysis tools makes it desirable to further accelerate the algorithm for large datasets. This is especially true if the bootstrap analysis is performed and multiple trees need to be calculated [12], [3]. Since the $O(N^3)$ complexity of the neighbor joining method is determined by the amount of operations per search step performed globally at each iteration to find an optimal pair ("neighbors to join"), accelerating the neighbor-joining method requires a smarter search methodology which avoids brute-force reevaluation of all possible pairs of points. Our interest in accelerating the neighbor-joining method is motivated by our ongoing efforts to develop and improve NCBI interactive analysis web tools, such as the NCBI Influenza Virus Resource [13], [14], where the neighbor-joining method is the default tree method. The goal is to enable bootstrap analysis for meaningful dataset, in a timeframe acceptable for interactive web tools. This paper describes an ongoing effort toward this goal.

Accelerating strategies for the neighbor-joining method have been proposed by several authors. The QuickJoin algorithm [15], [16] uses the quad-tree data structure to accelerate the search for optimal value of the goal function in the neighbor-joining algorithm, while still constructing the same tree as the original algorithm. The ClearCut algorithm [17], [18] implements the relaxed neighbor-joining approach. The algorithm does not search for a globally optimal pair, but selects a locally optimal pair (i, j) at each step, such that $\delta_{ij} = u_i + u_j - d_{ij}$ is maximum for both δ_{ik} and δ_{jk} for all other nodes k. In the Fast Neighbor Joining method [19] the goal function is not optimized globally, but is rather optimized over a set, called the "visible set". The algorithm is guaranteed to produce the same results as the neighbor-joining methods for an additive input.

In this paper we pursue the same goal as [16]: to accelerate the search for a pair of nodes to be join while constructing the same tree as the classical neighbor-joining algorithm. We arrange the candidate pairs of nodes in buckets according to the value of the NJ goal function (see Figure 1). When number of nodes is large, the value of the neighbor-joining goal function $\delta_{ij} = u_i + u_j - d_{ij}$ changes slightly with each iteration for each pair of nodes. As a result, only elements of the top bucket are re-evaluated at every iteration, while elements of lower buckets are accessed more rarely, when needed. Only a small portion

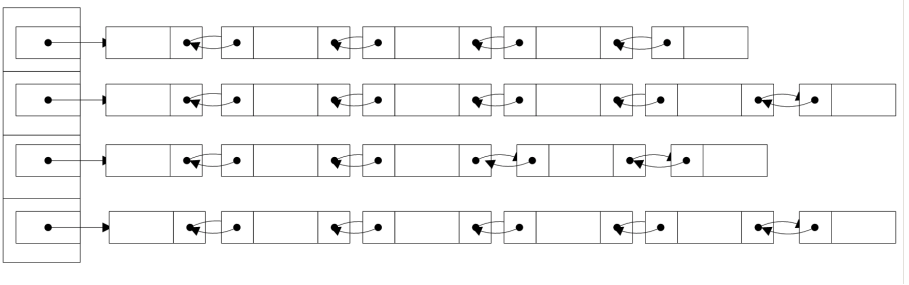


Fig. 1. A bucket data structure

of candidate pairs of nodes is examined at each iteration. $O(N)$ buckets are retained eliminating the need to evaluate $O(N^2)$ pairs per iteration. Instead, we rearrange $O(N)$ buckets without accessing the content and evaluate pairs in the top bucket. The proposed bucket-based data structure allows cache-efficient manipulations [20], [21]. Note that bucket data structures have been used in the bucket-sort algorithm [22] and shortest path algorithms in [23], [24], [25], [26], [27].

2 Methodology

Below we first describe the classical neighbor-joining algorithm and then show how to use the bucket data to perform an efficient search for a pair of nodes to be joined.

2.1 The Neighbor-Joining Method

Classical NJ algorithm. At each iteration $m = 0, ..., N-2$ of the classical neighbor-joining method [2],[3], average distances

$$u_i^m = \frac{1}{N-m-2} \sum_{j \neq i} D_{ij} \tag{1}$$

are calculated for each node. A global search is performed to find a pair of nodes (i_*, j_*) such that

$$(i_*, j_*) = \arg\max \delta_{ij}^m, \tag{2}$$

where

$$\delta_{ij}^m = u_i^m + u_j^m - D_{ij}. \tag{3}$$

Nodes i_* and j_* are joined in new node k_*.

The branch lengths v_{i_*} and v_{j_*} are calculated as

$$v_{i_*} = \frac{1}{2}(D_{i_* j_*} + u_{i_*}^m - u_{j_*}^m), \tag{4}$$

$$v_{j_*} = \frac{1}{2}(D_{i_* j_*} - u_{i_*}^m + u_{j_*}^m), \tag{5}$$

and distances from node k_* to the rest of the nodes are determined for each node p by the formula

$$D_{k_* p} = \frac{1}{2}(D_{i_* p} + D_{j_* p} - D_{i_* j_*}). \tag{6}$$

Preserving non-negativity of branch lengths and distances. For the implementations used in our web analysis tools [13], we chosen to keep branch lengths and distances non-negative. We modify formulas (4) and (5) as follows. Define

$$\gamma_{i_* j_*} = \frac{1}{2}\text{sign}(u_{i_*}^m - u_{j_*}^m) \min(|u_{i_*}^m - u_{j_*}^m|, D_{i_* j_*}). \tag{7}$$

Analogues of equations (4) and (5) are:

$$v_{i_*} = \frac{1}{2} D_{i_* j_*} + \gamma_{i_* j_*}, \tag{8}$$

$$v_{j_*} = \frac{1}{2} D_{i_* j_*} - \gamma_{i_* j_*}. \tag{9}$$

A non-negative analogue of equation (6) is

$$D_{k_* p} = \frac{1}{2} \max(D_{i_* p} + D_{j_* p} - D_{i_* j_*},\ 0). \tag{10}$$

Recursive formulas. New average distances u_i^m can be calculated from old ones u_i^{m-1} by formulas:

$$u_i^{m+1} = (1 + \frac{1}{N-m-3}) u_i^m - \frac{1}{((N-m-3))}(D_{ii_*} + D_{ij_*} - D_{ik_*})). \tag{11}$$

or, using (10),

$$u_i^{m+1} = (1 + \frac{1}{N-m-3}) u_i^m - \frac{1}{(N-m-3)} \min(D_{ii_*} + D_{ij_*}, \frac{1}{2}(D_{ii_*} + D_{ij_*} + D_{i_* j_*})). \tag{12}$$

2.2 Upper Bound for Change in the Goal Function Value for a Pair of Points over a Neighbor-Joining Step

Estimates for growth of average distances u_i. From (12), it is easy to obtain the following inequalities:

$$u_i^{m+1} \leq (1 + \frac{1}{N-m-3}) u_i^m. \tag{13}$$

and

$$u_i^{m+1} \leq u_i^m + \frac{1}{N-m-2} u_i^{m+1}. \tag{14}$$

Estimates for growth of δ_{ij}. From (3), it is easy to see that

$$\delta_{ij}^{m+1} - \delta_{ij}^m = (u_i^{m+1} - u_i^m) + (u_j^{m+1} - u_j^m).$$

Using the inequalities (13) and (14), we get

$$\delta_{ij}^{m+1} - \delta_{ij}^m \leq \frac{1}{N-m-3}(u_i^m + u_j^m).$$

or

$$\delta_{ij}^{m+1} - \delta_{ij}^m \leq \frac{1}{N-m-2}(u_i^{m+1} + u_j^{m+1}).$$

Finally, we obtain two growth estimates:

$$\delta_{ij}^{m+1} - \delta_{ij}^{m} \leq \Delta_{-}^{m}, \tag{15}$$

and

$$\delta_{ij}^{m+1} - \delta_{ij}^{m} \leq \Delta_{+}^{m}, \tag{16}$$

where

$$\Delta_{-}^{m} = \frac{2U^{m}}{N-m-3},$$

$$\Delta_{+}^{m} = \frac{2U^{m+1}}{N-m-2},$$

and

$$U^{s} = \max_{i} u_{i}^{s}.$$

The estimates (15) and (16) together can be written as:

$$\delta_{ij}^{m+1} - \delta_{ij}^{m} \leq \Delta^{m}, \tag{17}$$

where

$$\Delta^{m} = \min(\Delta_{-}^{m}, \Delta_{+}^{m}).$$

Note. These estimates show that when the number of nodes is large, value δ_{ij}^{m+1} can increase from δ_{ij}^{m} only slightly. For example, when $n = N - m$ is about 10^3, the increase is limited by approximately $2 \cdot 10^{-3}$ of the maximal average distance.

2.3 Construction of Buckets and Operating Them

The arrangement of pairs (i, j) in groups is performed according to the values of the neighbor-joining goal function δ_{ij} defined by formula (3). Our purpose is to limit evaluation of the individual pairs only to those that were close to optimal in the previous iteration and may become optimal at the current step.

First, the treatment of pairs with zero or near zero distances between nodes is considered. Lets introduce a special bucket for pairs (i, j) such that

$$D_{ij} < \epsilon \max_{lp} D_{lp},$$

where ϵ is a small number (ex., $\epsilon = 10^{-6}$). We join zero-distance and near-zero-distance pairs first, as we do in our current implementation of the classical neighbor joining algorithm [13].

Let us consider regular pairs. Define the bucket intervals as follows:

$$(+\infty,\ \alpha_0^m),\ \ (\alpha_0^m,\ \alpha_1^m),\ \ ...,\ \ (\alpha_{n-2}^m,\ \alpha_{n-1}^m),\ \ (\alpha_{N-1}^m, -\infty), \tag{18}$$

where

$$\alpha_0^m > \alpha_1^m > ... > \alpha_{N-2}^m > \alpha_{N-1}^m.$$

Our initial idea was to use intervals constant step Δ^m:

$$\alpha_{k+1}^m = \alpha_k^m + \Delta^m, \quad k = 0, 1, ..., N-2. \tag{19}$$

If parameter α_0^m satisfies the condition

$$\alpha_0^m \leq \delta_{\max}^m - \Delta^m, \tag{20}$$

where $\delta_{\max}^m = \max_{ij} \delta_{ij}^m$, only elements of the top bucket can have δ_{ij}^m equal to $\delta_{\max}^m$. All other buckets contain suboptimal elements. Moreover, because of the estimate (15), pairs that are currently in the buckets with $k \geq 2$ remain suboptimal in the next iteration.

At each neighbor-joining iteration, a new collection of N buckets is constructed according to (19). New pairs appearing at the current iteration are placed in the buckets accordingly. Contents of most of the existing buckets is placed into new buckets without being evaluated. First, the new bucket index k_{new} is determined by formula

$$k_{new} = [\frac{\delta_{\text{top}}^m + k \cdot \Delta^m - \delta_{\text{top}}^{m+1}}{\Delta^{m+1}}], \tag{21}$$

for each old bucket. If $k_{new} \geq 1$, the content of the kth old bucket with $k \geq 1$ is placed into the k_{new}th new bucket without any evaluation. However, if according to formula (21) $k_{new} \leq 0$, then each pair of nodes in the kth old bucket is evaluated and placed into an appropriate new bucket. The pairs of nodes in the top bucket are always evaluated for finding the optimal pair. In process of evaluation, a pair can be removed from the top bucket for two reasons:

- if the bucket contains a node that has already been eliminated, the pair of nodes is removed;
- if the pair has a low value of δ_{ij}^m, it is moved from the top bucket to a corresponding lower bucket.

Values α_0^m selected in each iteration should satisfy (20). In the initial step, for $m = 0$, the value $\alpha_0^0 = \delta_{\max}^0$ is used, since the maximal value is computed when all pairs of nodes are placed in buckets. In subsequent steps, the maximal δ_{ij}^m in the top bucket is taken as α_0^m.

However, this simple construction (19) would not be efficient if the values δ_{ij}^m are distributed non-homogeneously: many intervals may be empty, while others overpopulated (In an extreme case, one outlier pair may stay on the top, while all others are placed in the bottom interval). In order to overcome this problem, we propose an adaptive construction of buckets.

In the adaptive construction, the intervals for the initial step ($m = 0$) are defined as follows:

$$\alpha_{k+1/2}^0 = \alpha_k^0 - \Delta^0; \tag{22}$$

$$\alpha_{k+1}^0 = \min(\ \alpha_{k+1/2}^0,\ \ \max(\ \delta_{ij}^0 \mid \delta_{ij}^0 \leq \alpha_{k+1/2}^0\ )\ ), \tag{23}$$

for $k = 0, 1, 2, ..., N-2$. If there are no δ_s^0 below $\alpha_{k+1/2}^0$, we stop adding intervals. The values α_0^m are defined in the same way as before: $\alpha_0^0 = \delta_{\max}^0$, and subsequent values α_0^m are set to the maximal δ_{ij}^m in the top bucket. The formulas (22)-(23) guarantee that each bucket at the initial step ($m = 0$) is not empty. Otherwise, it has the same useful properties as (19).

In the subsequent iterations ($m = 1, 2, 3, ...$) the buckets are operated as follows:

- The maximal δ_{ij}^m in the top bucket is taken as α_0^m. The elements of the top bucket are evaluated as described above.
- Existing buckets are moved to the top by setting $\alpha_k^m = \alpha_k^{m-1} + \Delta^m$ for $k > 0$.
- If $\alpha_k^m \geq \alpha_0^m$, the value of α_k^m is refreshed by resetting it to the maximal δ_{ij}^m in the bucket.
- If the refreshed value $\alpha_k^m \geq \alpha_0^m - \Delta^m/2$, the bucket is merged with the top bucket;
- New pairs (k_*, p) are considered in descenting order and first are attempted to be placed in an existing bucket. If

$$\min(\alpha_k^m | \alpha_k^m \geq \delta_{k_* p}^m) - \delta_{k_* p}^m < \Delta^m, \tag{24}$$

  then the pair (k_*, p) is placed in the bucket providing the minimum value. Otherwise, a bucket is created. The same procedure is applied to the pairs moved from the top bucket to lower buckets.
- If number of buckets exceeds N, the top N buckets are taken and the remaining buckets are merged in the lowest bucket. Empty buckets, if any, are also removed.

Data structures. Below we briefly describe data structures for a `record`, a `bucket`, and a `bucket collection`.

`Record.` For a pair of nodes i and j ($i < j$), we keep a record consisting of two indices and the value of the distance between the nodes: $R = (i, j, D_{ij})$. There is no reason to save the actual value of the goal function δ_{ij} since it is changed at each algorithm step and cannot be reused. However, keeping the D_{ij} value in the record allows to avoid gathering these values from a large two-dimensional array and makes the algorithm more cache-optimal [20], [21].

`Bucket.` Each bucket contains a *linked list* of records. In our initial implementation, we use the C++ STL `List` class. A standard constant-time `splice` algorithm [28] is used to combine link lists. Records referring to nodes which have already have been eliminated are erased using a C++ STL constant-time `List erase()` function. Special memory allocation and reallocation can be used to provide cache-efficient placement of bucket elements [29].

`Bucket Collection.` Each bucket collection contains two arrays: the first contains real numbers in decreasing order and describes bucket intervals, while the second contains pointers to buckets. In our initial implementation, we use C++ STL `vector` class for these arrays. As described above, N buckets are allocated at each neighbor-joining step.

In addition to bucket-based data structures we use arrays implemented as C++ STL `vector` objects for u_i^m, and for describing status of the node (active/ non-active) that is changed when node is eliminated. Since calculating new distances by formula (6) requires direct access to D_{ij}, we keep this two-dimensional matrix as `vector< vector<double> >` in our initial implementation.

3 Test Results

To evaluate the algorithm, we used the following four data sets containing full-length Influenza A H3N2 hemagglutinin protein coding sequences obtained from the NCBI Influenza Virus Resource [13]:

- Dataset containing 44 sequences from 1968-1972;
- Dataset containing 252 sequences from 1971-1996;
- Dataset containing 504 sequences from 1972-2000;
- Dataset containing 1000 sequences from 1972-2005.

Figure 2 illustates the performance of the algorithm for these 4 datasets. The total number of pair evaluations stays approximately proportional to N^2, where N is the number of sequences in the dataset, with a constant $c \approx 3$ for $N = 252$, $N = 504$ and $N = 1000$, while $c \approx 7.5$ for $N = 42$.

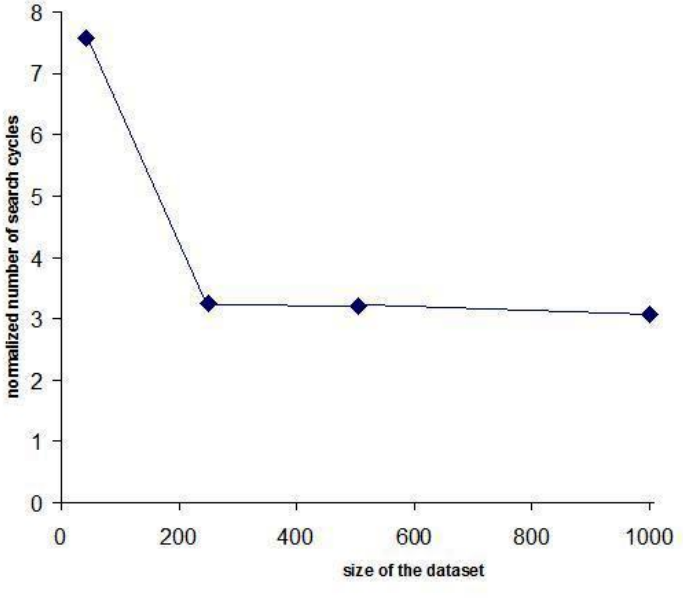


Fig. 2. Number of pair evaluations per cycle divided by N^2

Figure 3 shows the initial distribution of $\delta(i,j) = u(i) + u(j) - d(i,j)$ for non-identical pairs of sequences in the largest dataset (e.g., $N = 1000$). The maximal value is 0.441. Only 100 pairs of sequences out of $499,500$ (0.02%) have values of $\delta(i,j)$ greater than 0.3, and only about 1,000 pairs of sequences (0.2%) have values of $\delta(i,j)$ above 0.1.

Figure 4 shows the number of pair evaluations in each iteration of the algorithm. The total number of pair evaluations is accumulated from a very small

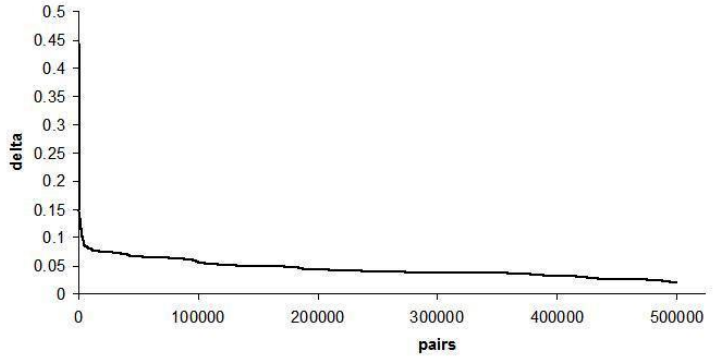


Fig. 3. Values of $u(i) + u(j) - d(i, j)$ for pairs of sequences, $N = 1000$

Table 1. Normalized execution times

Number of sequences	NCBI gizmo2 computer	NCBI sutils0 computer
252	$6.3 \cdot 10^{-7}$	$1.1 \cdot 10^{-6}$
504	$9.1 \cdot 10^{-7}$	$1.61 \cdot 10^{-6}$
1000	$1.33 \cdot 10^{-6}$	$2.34 \cdot 10^{-6}$

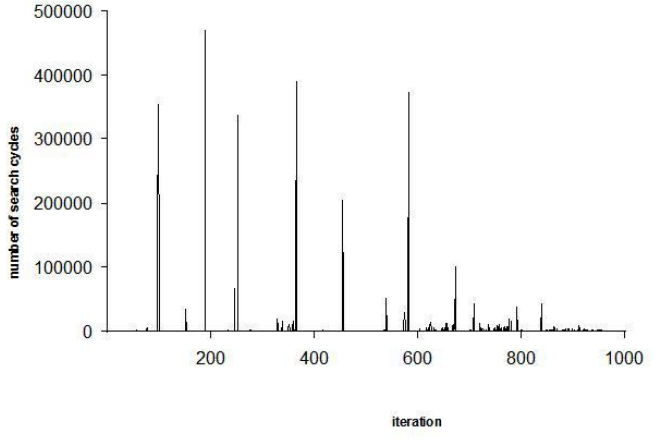


Fig. 4. Number of pair evaluations in each iteration

number of evaluations in the majority of steps and a large number of evaluations on rare occasions.

Figure 5 shows the execution times for the tests on two NCBI Linux computers: gizmo2 - a computer having an Intel Xeon E5345 2.33 GHz processor with

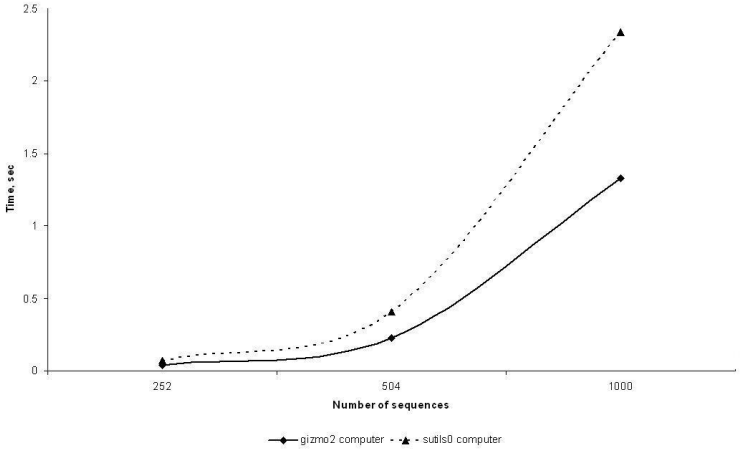


Fig. 5. Execution times

4MB cache size, and sutils0 - a computer having an Intel Xeon 3.20 GHz processor with 512 KB cache size. Corresponding execution times divided by N^2, where N is number of sequences in the test, are shown in the table below. Note that while sutils0 has a faster processor, gizmo2 consistently demonstrates 1.7 - 1.8 times faster execution time, probably because its cache size is much larger. We believe that the speed of memory access rather than processor speed is critical for the speed of execution, and the code will greatly benefit from tuning aimed to minimize cache misses. Comparison of our normalized times $1.33 \cdot 10^{-6}$ seconds and $2.34 \cdot 10^{-6}$ seconds to $5.64 \cdot 10^{-6}$ seconds reported for QuickJoin (based on Table 1 in [15]), allows us to be very optimistic about the performance of an optimized version of the code.

4 Discussion

Accelerating the neighbor-joining method is important for enhancing performance of the online web analysis tools, where users expect to perform initial exploratory analysis of the datasets in real time and perform bootstrapping as fast as possible. In this paper we present an adaptive bucket algorithm able to significantly reduce the amount of evaluations in search steps by distributing candidate pairs in buckets and evaluating a small portion of all pairs in each iteration. The proposed construction helps to avoid empty buckets and allows the algorithm to handle the values of the neighbor-joining goal function which are distributed non-homogeneously, including cases when outliers are present. The algorithm uses simple data structures that can be further optimized, including optimizing the cache-efficiency. Our the preliminary test results are shown above, and we continue optimizing the code and plan to perform comprehensive comparisons. While the proposed algorithm is designed to produce the same results

as classical neighbor-joining, the degree of acceleration it provides is determined by the distribution of the values of the neighbor-joining goal function that, in turn, depends on the structure of the dataset.

Acknowledgements

This research was supported by the Intramural Research Program of the NIH, National Library of Medicine.

The authors are thankful to David J. Lipman, Alejandro Schaffer, Stacy Ciufo, Vahan Grigoryan and Yuri Kapustin for productive discussions.

References

1. Saitau, N., Nei, M.: The neighbor-joining method: new method for reconstructing phylogenetic trees. Mol. Biol. Evol. 4, 406–425 (1987)
2. Studier, J.A., Keppler, K.J.: A note on the neighbor-joining algorithm of Saitou and Nei. Molecular Biology and Evolution 5, 729–731 (1988)
3. Felsenstein, J.: Inferring Phylogenies. Cambridge University Press, Cambridge (2003)
4. Atteson, K.: The performance of neighbor-joining methods of phylogenetic reconstruction. Algorithmica 25, 251–278 (1999)
5. Tamura, K., Nei, M., Kumar, S.: Prospects for inferring very large phylogenies by using the neighbor-joining method. PNAS 101(30), 11030–11035 (2004)
6. Bryant, D.: On the uniqueness of the selection criterion in neighbor-joining. Journal of Classification 22(1) (2005)
7. Desper, R., Gascuel, O.: The minimum-evolution distance-based approach to phylogenetic interference. In: Gascuel, O. (ed.) Mathematics of evolution and phylogeny, pp. 1–32. Oxford University Press, Oxford (2005)
8. Gascuel, O., Steel, M.: Neighbor-joining revealed. Molecular Biology and Evolution 23, 1997–2000 (2006)
9. Gascuel, O.: BIONJ: an improved version of the nj algorithm based on a simple model of sequence data. Mol. Biol. Evol. 14, 685–695 (1997)
10. Bruno, W.J., Socci, N., Halpern, A.L.: Weighted neighbor-joining: a likelihood-based approach to distance-based phyloginy reconstruction. Mol. Biol. Evol. 17, 189–197 (2000)
11. Yang, Z.: Computational Molecular Evolution. Oxford University Press, Oxford (2006)
12. Bryant, D.: A classification of consensus methods for phylogenies. In: Janowitz, M., Lapointe, F.J., McMorris, F., Mirkin, B., Roberts, F., (eds.) BioConsensus, DIMACS, Americal Mathematical Society, pp. 163–184 (2003)
13. Bao, Y., Bolotov, P., Dernovoy, D., Kiryutin, B., Zaslavsky, L., Tatusova, T., Ostell, J., Lipman, D.: The Influenza Virus Resource at the National Center for Biotechnology Information. Journal of Virology 82(2), 596–601 (2008)
14. Zaslavsky, L., Bao, Y., Tatusova, T.A.: An Adaptive-Resolution Tree Visualization of Large Influenza Virus Sequence Datasets. In: Măndoiu, I.I., Zelikovsky, A. (eds.) ISBRA 2007. LNCS (LNBI), vol. 4463, pp. 192–202. Springer, Heidelberg (2007)
15. Mailund, T., Pedersen, C.N.: Quickjoin – fast neighbor-joining tree reconstruction. Bioinformatics 20(17), 3261–3262 (2004)

16. Mailund, T., Brodal, G.S., Fagerberg, R., Pedersen, C.N.S., Phillips, D.: Recrafting the neighbor-joining method. BMC Bioinformatics 7(29) (2006)
17. Shenerman, L., Evans, J., Foster, J.A.: Clearcut: fast implementation of relaxed neighbor joining. Bioinformatics 22(22), 2823–2824 (2006)
18. Evans, J., Shenerman, L., Foster, J.: Relaxed Neighbor-Joining: A Fast Distance-Based Phylogenetic Tree Construction Method. J. Mol. Evol. 62, 785–792 (2006)
19. Elias, I., Lagergren, J.: Fast neighbor joining. In: Caires, L., Italiano, G.F., Monteiro, L., Palamidessi, C., Yung, M. (eds.) ICALP 2005. LNCS, vol. 3580, pp. 1263–1274. Springer, Heidelberg (2005)
20. LaMarca, A., Ladner, R.E.: The influence of caches on the performance of sorting. Journal of Algorithms 31, 66–104 (1999)
21. Brodal, G.S., Fagerberg, R., Vinther, K.: Engineering a cache-oblivious sorting algorithm. Journal of Experimental Algorithmics 12, 2.1 (2007)
22. Cormen, T.H., Leiserson, C.E., Rivest, R.L., Stein, C.: Introduction to Algorithms, 2nd edn. MIT Press and McGraw-Hill (2001)
23. Dial, R.B.: Algorithm 360: Shortest path forest with topological ordering. Comm. ACM 12, 632–633 (1969)
24. Wagner, R.A.: A shortest path algorithm for edge-aparse graphs. J. Assoc. Comput. Mach. 23, 50–57 (1976)
25. Dinic, E.A.: Economical algorithms for finding shortest path in network. In: Popkov, Y.S., Shmulyan, B.L., (eds.) Transportation Modeling Systems, The Institute for System Studies, pp. 36–44 (in Russian) (1978)
26. Denardo, E.V., Fox, B.L.: Shortest-route methods: 1. reaching, pruning, and buckets. Oper. Res. 27, 161–186 (1979)
27. Cherkassky, B.V., Goldberg, A.V., Silverstein, C.: Buckets, heaps, lists, and monotone priority queues. SIAM Journal of Computing 1999 28(4), 1326–1346 (1999)
28. Musser, D.R., Derge, G.J., Saini, A.: STL Tutorial and Reference Guide: C++ Programming with the Standard Template Library, 2nd edn. Addison-Wesley, Reading (2001)
29. Meyers, S.: Effective STL. Addison-Wesley, Reading (2001)

Generalized Gene Adjacencies, Graph Bandwidth and Clusters in Yeast Evolution

Qian Zhu[1], Zaky Adam[1], Vicky Choi[2], and David Sankoff[1]

[1] Department of Biochemistry, School of Information Technology and Engineering, and Department of Mathematics and Statistics, University of Ottawa, Ottawa, Canada K1N 6N5
{qzhu012,zadam008,sankoff}@uottawa.ca
[2] Department of Computer Science, Virginia Tech., Blacksburg, VA 24061
vchoi@cs.vt.edu

Abstract. We present a parametrized definition of gene clusters that allows us to control the emphasis placed on conserved order within a cluster. Though motivated by biological rather than mathematical considerations, this parameter turns out to be closely related to the maximum bandwidth parameter of a graph. Our focus will be on how this parameter affects the characteristics of clusters: how numerous they are, how large they are, how rearranged they are and to what extent they are preserved from ancestor to descendant in a phylogenetic tree. We infer the latter property by dynamic programming optimization of the presence of individual edges at the ancestral nodes of the phylogeny. We apply our analysis to a set of genomes drawn from the Yeast Gene Order Browser.

1 Introduction

The definition of synteny blocks, gene clusters or similar constructs from the comparison of two or more genomes entails a trade-off of great consequence: if we place emphasis on identical content and order of the genes, segments or markers in a block or cluster, only relatively small regions of the genome will satisfy this restrictive condition, giving rise to a plethora of tiny blocks while missing large regions common to the genomes. On the other hand, by allowing unrestricted scrambling of genes within blocks (e.g., max-gap [1] or "gene teams" [7]), we forgo accounting for local genome rearrangement, missing an important aspect of evolutionary history, or we relinquish the possibility of pinpointing extensive local conservation, where this exists.

In this paper, we present a parametrized definition of gene clusters that allows us to control the emphasis placed on conserved order within a cluster. Though motivated by biological rather than mathematical considerations, this parameter turns out to be closely related to the maximum bandwidth parameter of a graph. Our focus will be on how this parameter affects the characteristics of clusters: how numerous they are, how large they are, how rearranged they are and to

I. Măndoiu, R. Sunderraman, and A. Zelikovsky (Eds.): ISBRA 2008, LNBI 4983, pp. 134–145, 2008.

what extent they are preserved from ancestor to descendant in a phylogenetic tree. We infer the latter property by dynamic programming optimization of the presence of individual edges in a generalized adjacency graph abstractly representing chromosomal gene order. We apply our analysis to a set of genomes drawn from the Yeast Gene Order Browser (YGOB) [3]. Among the results, we find strong evidence for setting a certain fixed value to the cluster parameter. We also find that we can recover almost all the clusters that can be found without order constraints, i.e., by the max-gap criterion, indicating that local order conservation is a lot greater than that unconstrained definition would suggest.

2 Definitions

Our characterization of gene clusters is made up of a general part that identifies clusters of vertices common to two graphs, and a specific part where a graph is determined by the proximity of genes on the chromosomes of a genome. This is illustrated in Figure 1.

Definition 1. *Let $G_S = (V_S, E_S)$ and $G_T = (V_T, E_T)$ be two graphs with a non-null set of vertices in common $V = V_S \cap V_T$. We say a subset of $C \subseteq V$ is an ST-***cluster** *if it consists of the vertices of a maximal connected subgraph of $G_{ST} = (V, E_S \cap E_T)$.*

Definition 2. *For the purposes of genome comparison, we may consider V_X to be the set of genes in the genome X. For genes g and h in V_X on the same chromosome in X, let $gh \in E_X$ if the number of genes intervening between g and h in X is less than θ, where $\theta \geq 1$ is a fixed* **neighbourhood parameter**.

These definitions of edge sets and ST-clusters decompose the genes in the two genomes into identical sets of disjoint clusters of size greater or equal to 2, and possibly different sets of singletons belonging to no cluster, either because they are in V, but not in $E_S \cap E_T$, or because they are in $(V_S \cup V_T \setminus V)$. For simplicity, we do not attempt to deal with duplicate genes in this paper. When $\theta = 1$, a cluster has exactly the same gene content and order (or reversed order) in both genomes. When $\theta = \infty$, the definition returns simply all the synteny sets, namely the sets of genes in common between two chromosomes, one in each genome.

Let Π be the set of all orderings of V. Recall that the **bandwidth** of a graph $G(V, E)$ is defined to be

$$B(G) = \min_{p \in \Pi} \max_{uv \in E} |p(u) - p(v)|. \tag{1}$$

In a genome S each chromosome χ determines a physical order among the genes it contains.

Proposition 1. *Bandwidth $B(G_S) = \theta$, as long there are at least $2\theta + 1$ genes on some chromosome χ in genome S.*

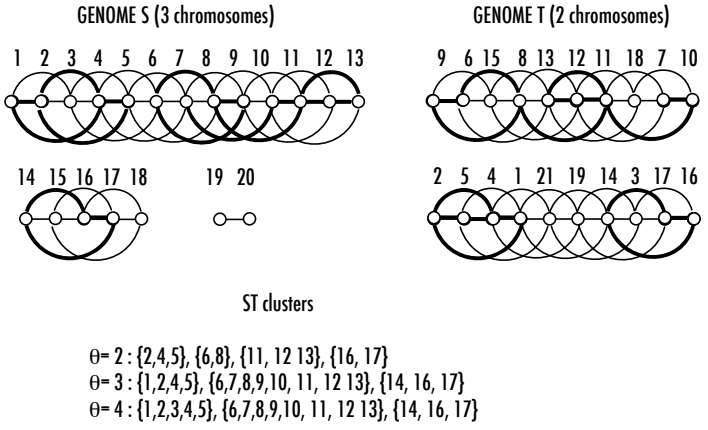


Fig. 1. Graphs constructed from two genomes using parameter $\theta = 3$. Thick edges determine clusters. ST-clusters listed for $\theta = 2$ and $\theta = 4$ as well.

Proof: By Definition 2, the vertex v corresponding to the gene at position $\theta+1$ on chromosome χ is connected to 2θ other vertices. The most remote of these are at positions 1 and $2\theta+1$. Similarly, for a vertex u at any other position on χ, we can show that the most remote gene connected to u is no farther away than θ. Thus, for the order $p(\cdot)$ on the vertices defined by the original gene order, $\max|p(u) - p(v)| = \theta$. Hence, $B(G_S) \leq \theta$.

For any other order $p(\cdot)$, consider the 2θ vertices connected to vertex v. For one such vertex w, $|p(v) - p(w)| \geq \theta$, since we cannot fit 2θ vertices connected to v into an interval of size $< 2\theta + 1$, also containing v.

Since the upper and lower bounds coincide, the proposition follows. □

Proposition 2

$$B(G_{ST}) = \max_{C \in \mathcal{C}} B(C), \tag{2}$$

where $\mathcal{C}$ is the set of connected components of G_{ST}.

Proof: Since E_{ST} is the union of the edges in all the C,

$$\max_{uv \in E_{ST}} |p(u) - p(v)| = \max_{C \in \mathcal{C}} \max_{uv \in E_C} |\bar{p}(u) - \bar{p}(v)|, \tag{3}$$

where $\bar{p}(\cdot)$ is the order induced on the vertices in C by the order $p(\cdot)$ on E_{ST}. But any set of vertex orders on all the individual C can be jointly extended to an order on V_{ST}. □

We compare the definition of an ST-cluster with that of a **max-gap cluster** [1,7].

Definition 3. *Let $\theta \geq 1$. Let $V_C \subseteq V_S \cap V_T$ be a set of r vertices corresponding to genes all on the same chromosome χ_S in genome S and all on the same*

chromosome χ_T in genome T. Let $g_1, g_2, \ldots, g_r$ be a labelling of these genes according to their order on χ_S. Let $h_1, h_2, \ldots, h_r$ be a labelling of these same genes according to their order on χ_T. Let $p_S(\cdot)$ and $p_T(\cdot)$ indicate the positions of genes on χ_S and χ_T, respectively. Then if

$$|p_S(g_{i+1}) - p_S(g_i)| \leq \theta \text{ and } |p_T(h_{i+1}) - p_T(h_i)| \leq \theta \tag{4}$$

for all $1 \leq i \leq r-1$, then V_C satisfies the max-gap criterion. If, in addition V_C is contained in no larger V_F also satisfying the criterion, then V_C is said to be a max-gap cluster.

Proposition 3. *Every ST-cluster with parameter θ satisfies the max-gap criterion with the same value of θ.*

Proof: Consider two successive genes in the ST-cluster in genome S. By Definition 2, they cannot be separated by more than $\theta - 1$ genes not in the cluster. Since this holds for all pairs of successive genes, both in S and in T, the max-gap criterion is met. □

The converse of Proposition 3 does not hold, however. The max-gap criterion only limits the number of **non-cluster elements** intervening, in either genome, between two cluster elements. Thus in the max-gap definition with $\theta = 2$, we could have a cluster $\{a, b, c, d, e, f\}$ with order $abcdef$ in S and $fbcdea$ in T, but this would not be an ST-cluster (though $\{b, c, d, e\}$ would be). Also, $a * bc$ in S and $bc * a$ in G_T could define a max-gap cluster $\{a, b, c\}$, where the asterisks represent genes not present, or remote, in S and T, respectively, but this would not be a ST-cluster (though $\{b, c\}$ would be).

3 Comparisons of Yeast Genomes

The data. The Yeast Gene Order Browser (YGOB) [3] contains complete gene orders and orthology identification among the five yeast species depicted in Figure 2: two descendents of an ancient genome duplication event, *Saccharomyces cerevisiae* and *Candida glabrata*, and three species that diverged before this event, *Ashbya gossypii, Kluyveromyces waltii* and *Kluyveromyces lactis.* For the ancient tetraploids, YGOB includes a reconstruction of the ancestral genome, which, with the help of further details supplied by Kevin Byrne and Jonathan Gordon (personal communication), allows us to identify duplicate genes as belonging to one of the two ancestral lineages, indicated by A and B in the figure, and to find two complete sets of clusters in each of these species, one in each lineage. For our purposes, then, the duplicate lineage effectively expands the data set from five to seven genomes.

Notation. With reference to Fig. 2 we will refer to the common ancestor of *Ashbya gossypii* and *Kluyveromyces lactis* as Node D, and to its immediate ancestor as Y. Nodes A and B will refer to the two ancestral lineages within both

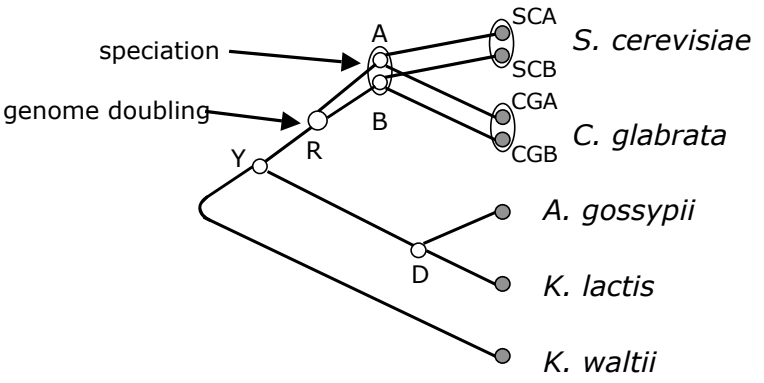


Fig. 2. Phylogeny of yeasts in YGOB. Whole genome doubling event at R giving rise to A and B lineages in *S. cerevisiae* (SCA, SCB) and *C. glabrata* (CGA, CGB) indicated, as is the speciation event at the divergence of these two species. Choice among the identified ancestor nodes Y, R, D, A or B to be the root is arbitrary in our mathematical analysis, but historically, the earliest divergence time is represented by the branching at the left of the phylogeny.

Saccharomyces cerevisiae and *Candida glabrata* at the time of speciation, while Node R will designate the tetraploid ancestral to these.

Defining lineage-specific clusters within a tetraploid descendant. The YGOB indicates the common ancestry, pre-speciation, in *Saccharomyces cerevisiae* and *Candida glabrata*, of two separate gene lineages, labelled A and B, in both genomes. To apply Definition 2, we first masked the identity of all lineage B genes without deleting them from their positions, and then applied the criterion to the lineage A genes to produce the edges in G_{SCA} and G_{CGA}. We then reversed roles of A and B, masking the identity of all lineage A genes without deleting them from their positions, and then applied the criterion to the lineage B genes to obtain G_{SCB} and G_{CGB}. Fig. 3 shows plots of the number of clusters detected as a function of θ, decreasing as a result of cluster amalgamation, featuring a distinct elbow near $\theta = 3$ for all the pairwise comparisons. This also shows a striking resemblance to the same analysis for max-gap clusters, suggesting that in these data, the max-gap clusters also satisfy our more stringent generalized adjacency criterion. In other contexts, perhaps in prokaryotes, more intense processes of local gene rearrangement may result in relatively more max-gap clusters.

In Fig. 3, we depict how cluster size is distributed and use this to assess the degree of relatedness of genomes or lineages.

4 Extensions to m Genomes

We can extend the definition of an ST-cluster based on two genomes S and T to an $ST\cdots U$-cluster based on the graphs $G_S, G_T, \ldots, G_U$ induced by the m

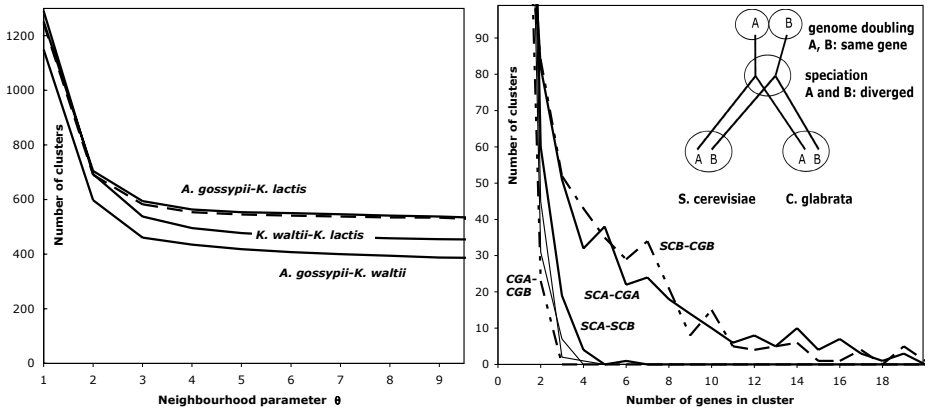


Fig. 3. Left: Dependence of number of clusters on neighbourhood parameter, showing that, independent of θ, *K. waltii* has fewer (larger) clusters when compared with the other two genomes, as might be expected from the closer phylogenetic relationship of the latter in Fig. 2. Dashed line indicates that the max-gap criterion returns fewer, larger clusters for the same value of θ — one max-gap cluster may contain several ST-clusters. Downward slope of all lines due to the incorporation of smaller clusters into larger ones as θ increases, demonstrating that almost all max-gap clusters also have conserved neighbourhood structure. Right: Distribution of size of clusters for $\theta = 2$, showing larger clusters, i.e., less evolutionary divergence, between same lineage SCA-CGA and SCB-CGB in different species than between different lineages. Also, the two different lineages are more diverged in CG than in SC, as confirmed for larger θ (not shown), consistent with the highly derived nature of the *C. glabrata* genome. Two thinner, unlabeled curves indicate SCA-CGB and SCB-CGA.

genomes $S, T, \ldots, U$. We simply extend Definition 1 to involve the intersection of the edge sets of m graphs instead of 2 graphs,

$$G_{ST\cdots U} = (V_S \cap V_T \cap \cdots \cap V_U, E_S \cap E_T \cap \cdots \cap E_U) \tag{5}$$

and then retain the set of vertices in each of the connected components of this graph as the $ST\cdots U$-clusters.

A more useful generalization turns out to involve the **median** of the m genomes $M_{ST\cdots U} = (V_S \cup V_T \cup \cdots \cup V_U, E)$, where E minimizes the sum of the sizes of the symmetric differences between E and the E_X. This is rapidly calculated using a majority rule. This graph may, however, sometimes not correspond to any genome, as we will discuss in Section 7. To verify that it does, we have to solve the fixed parameter version of the maximum bandwidth problem, which has a polynomial (but hard to implement) dynamic programming solution. Otherwise we can try to find the largest subset of E with bandwidth $\leq \theta$, which may require exponential search.

5 Optimizing Ancestral Nodes Minimizing Edge Appearances/Disappearances

Consider that the data at each terminal node consist of zeroes or ones, representing the presence or absence of each edge ϵ in that data genome. We wish to assign a zero or one for each edge at each ancestral node so as to minimize the number of times the presence/absence indicator changes value from one endpoint of an edge to the other, summed over all branches in the tree and summed over all edges ϵ. We will discuss this ancestral node optimization for unrooted binary trees, i.e., where each ancestral node has exactly three adjacent nodes, perhaps the simplest instance of dynamic programming on a tree [5, Chapter 2]. (This procedure is easily extended to non-binary trees.)

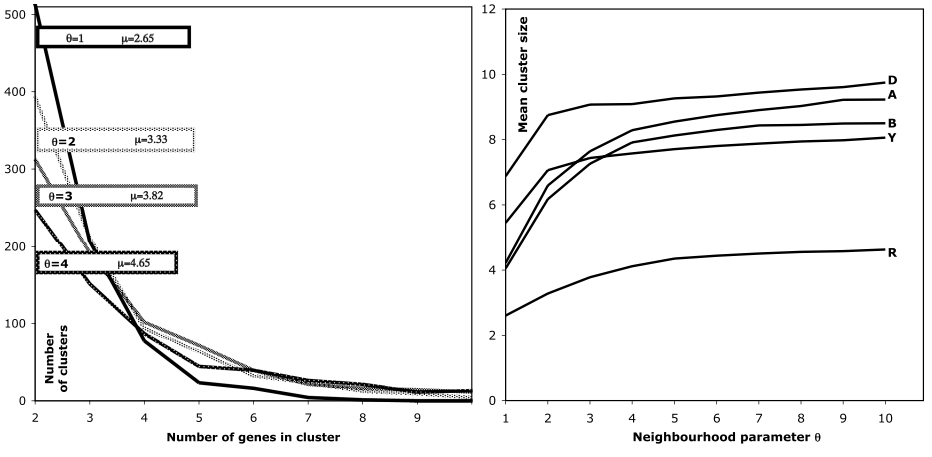


Fig. 4. Left: Distributions of cluster size, with mean μ, at Node R, for various values of θ. Smaller clusters amalgamate into larger ones as θ increases. Right: Mean cluster size at ancestral nodes, for various values of θ.

Dynamic programming requires two passes. In the forward pass, from the terminal nodes towards the root R (chosen arbitrarily from among the ancestor nodes, without consequences for the results), the value of the variable (the presence or absence of ϵ) may be established definitely at some ancestral nodes, while at other nodes it is left unresolved until the second, "traceback" pass, when any multiple solutions are also identified. We call those edges that are definitely present at a node the *optimals*, while those that are potentially present during the forward pass, the *near-optimals*. We need not discuss further those that are definitely excluded during the forward pass.

Note that the (arbitrary) designation of one ancestor node to be the root R determines, for each branch, which of its endpoints corresponds to the mother genome (the one proximal to the root), and which to the daughter (the one distal

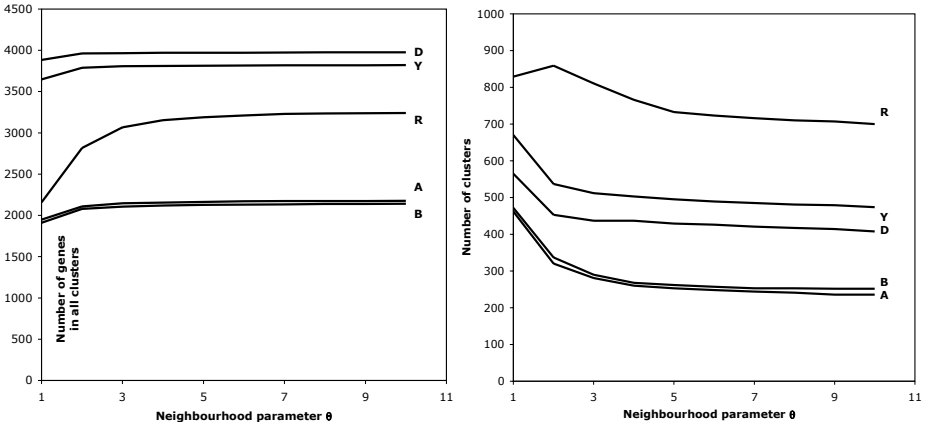


Fig. 5. Left: Total number of genes in clusters is remarkably stable, except for Node R, which recruits more genes up to $\theta = 4$. Right: As θ increases small clusters are amalgamated with larger ones, so that the total number of clusters decreases.

to the root). We order the nodes so that no node precedes any of its daughters. (This is always possible for a rooted tree.)

Suppose ancestral node N (other than the root R) has daughter nodes K and H. Because of the way we have ordered the nodes, by the time we reach N during the forward pass, we have already decided, for each daughter, whether ϵ is an optimal or near-optimal. Then if ϵ is optimal for both K and H, then it is optimal for N. If it is optimal for only one of K and H, it is near-optimal for N. For the root node R, with three daughters, if ϵ is optimal for at least two of the three, then it is optimal for R. We need not consider near-optimals for R.

For the traceback, reversing direction in the same order, starting at R, if ϵ is optimal for a mother node and near-optimal for its daughter, then ϵ is promoted to optimal status in the daughter. (This operationalizes the "majority rule" mentioned in Section 4.)

Note that in this method, the presence or absence of genes in the ancestral genomes derives solely from the presence or absence of at least one edge having that gene as an endpoint.

6 Gene Clusters at the Ancestral Nodes of the Yeast Phylogeny

6.1 Cluster Statistics

Introducing the generalized adjacencies through the neighbourhood parameter allows clusters to be conserved despite local rearrangements. This is seen in Fig. 4, where the distribution of cluster sizes (number of vertices) at Node R is seen to spread out to larger values as θ increases.

The average sizes of clusters is much higher in the other ancestral nodes, though they follow the same trend, as is seen on the right of Fig. 4.

While the average cluster size increases, the number of genes involved in these clusters at a given node does not change much, as seen in Fig. 5. Consequently, as seen on the right of the figure, the number of clusters drops.

6.2 Evolution and Cluster Coherency

From node to node the number of clusters and the genes they contain change. We can, however, assess to what extent this change is gradual or abrupt. If a cluster simply gains or loses a few genes, or if a cluster divides in two, or if two merge to become one, we may consider the resulting configuration a gradual change. We operationalize this by saying two clusters, one in each of two ancestral genomes, are in conflict unless one is nested in the other or they are disjoint. In Table 1, we show what proportion of each ancestor's clusters are in conflict with their adjacent nodes' clusters.

Table 1. Conflicts in clusters beteeen genomes at two ends of each tree branch, as a function of θ. Percentage conflict out of the total number of clusters in genome in left hand column.

		Neighbourhood parameter		
Node	Adjacent Node	1	3	8
A	R	20	36	37
B	R	23	36	40
R	A	10	16	16
R	B	11	16	17
R	Y	0	0	0
D	Y	0	1	1
Y	D	0	0	1
Y	R	0	1	1

Thus cluster evolution has been exceedingly gradual among the diploid genomes, but a good proportion of the A and B lineage clusters are seriously disrupted in their common ancestor.

7 Bandwidth of the Clusters

We have constructed clusters of genes based on adjacencies presumed to have been present in the ancestral genomes. While these are most parsimonious inferences, they are not sufficient to reconstruct the entire genomes, mainly because we have tried to compute neither how to partition the clusters among chromosomes nor how to impose a linear order within a cluster. Indeed, the dynamic programming is not even able to ensure that the clusters are compatible with the

generalized adjacency structure imposed on the data genomes in Definition 2, for the reasons alluded to in Section 4. In other words, there is no constraint on the connected components, and hence the entire graph inferred at an ancestral node, to have maximum bandwidth $\leq \theta$. If the bandwidth is larger, it means that we can construct no genome where the vertices in the connected component in question can be linearly disposed so that each edge has less than θ genes intervening between the two endpoints.

On the other hand, there is no compelling reason to insist on this bandwidth restriction on the ancestral genomes. Our initial goal was to find how clusters of vertices are preserved or evolve along various evolutionary lineages, and if the bandwidth is larger at some ancestor, this simply suggests that the cluster was looser at that time.

Whatever the importance or the interpretation we attach to bandwidth, it is thus of great importance to see how it is preserved or changed in the ancestral genomes we are investigating.

The problem of inferring the maximum bandwidth of a graph is, however, not trivial. Indeed, it is NP-complete [9], though Saxe [10] showed that detecting whether bandwidth is greater than θ is of polynomial complexity. Unfortunately, we have no implementation of Saxe's dauntingly high-level pseudo-code; in any case we are more interested in knowing the bandwidth than in testing whether it is greater than θ.

Thus we are led to investigate the many heuristics available for estimating the bandwidth. For example if there is a vertex of degree $> 2\psi$, the bandwidth must be greater than ψ, as is clear from the upper bound discussed in Theorem 1. Many heuristics emanate from an interest in reducing bandwidth in matrix theory. For sparse matrices, the best-known method is the Cuthill-McKee method [4] and its modification, the reverse Cuthill-McKee (RCM) algorithm [6,8]. We have implemented the latter to study the bandwidth of the components we have reconstructed at the ancestral nodes. Since the results of RCM depend on the input order of the vertices, we ran the algorithm 100 times with different orders to find the minimum estimate for the bandwidth, as displayed in Table 2. In all but three of the thirty entries in the table, the value shown was already detected after 10 runs.

As can be seen, the bandwidth exceeds θ in most cases. Inspection of the graphs show that this is due to a small number of vertices of high degree. If we wanted to constrain the ancestral genome graphs to have maximum bandwidth $\leq \theta$, we could:

- After the dynamic programming, test each node for bandwidth and exclude one or more vertices from the graph. It would not seem appropriate, however, to exclude the vertices of highest degree, since these are likely to be the most central to the cluster. Rather we would exclude some vertices of low degree adjacent to vertices of highest degree.
- As a less *ad hoc* solution, during the traceback of the dynamic programming, ensure that the set of edges being constructed never exceeds bandwidth θ. This may be a complex undertaking, however, since it may require testing all subsets of the set of near optimal edges eligible to promotion to optimal

Table 2. Minimum out of 100 runs of the RCM algorithm applied to edge sets produced by dynamic programming at the ancestral nodes of yeast evolutionary tree for various values of the neighbourhood parameter θ.

θ	Node				
	A	R	B	Y	D
1	1	1	1	1	1
2	3	4	2	3	2
3	4	5	4	5	5
4	5	7	5	7	5
5	6	7	6	8	7
6	9	8	7	8	8
7	9	10	9	9	9
8	9	11	10	11	11
9	13	13	11	10	11
10	13	13	11	16	17

status. Note that this approach potentially interferes with the exactness of the dynamic programming.

– Intervene directly in the dynamic programming recurrence. To conserve exactness, this approach would require storing and searching over a structure more complex than just the sets of optimals and near optimals.

8 Conclusions

The generalized adjacencies we have introduced allow us to recognize clusters even though they have been perturbed by local rearrangements. That the max-gap criterion gives approximately the same number of clusters means that max-gap is too weak a criterion in this context in that it doesn't recognize order conservation in the clusters.

Our separation of the A and B lineages as separate phylogenetic lineages is validated by the higher number of within-lineage clusters than within-species clusters, with the *C. glabrata* genome appearing highly rearranged.

We have shown the interplay of bandwidth considerations and the dynamic programming optimization of ancestral nodes in a given phylogeny. There is scope for improving our estimate of bandwidth, perhaps with approximation algorithms such as the semi-definite-programming approach in [2].

The neighbourhood parameter allows us to control the distribution of cluster sizes and the number of clusters. It allows us to explore the trade-off between the size of clusters and the rate of conflict between clusters in connected ancestral nodes.

Acknowledgments

We thank Ken Wolfe, Kevin Byrne and Jonathan Gordon for encouragement and for valuable information, including key data beyond the inestimable resource

they have made available to the yeast and genomics communities. We also thank Howard Bussey for a helpful discussion and Robert Warren for his sustained collaboration in the polyploid genomics project. Research supported in part by a grant to DS from the Natural Sciences and Engineering Research Council of Canada (NSERC). DS holds the Canada Research Chair in Mathematical Genomics.

References

1. Bergeron, A., Corteel, S., Raffinot, M.: The algorithmic of gene teams. In: Guigó, R., Gusfield, D. (eds.) WABI 2002. LNCS, vol. 2452, pp. 464–476. Springer, Heidelberg (2002)
2. Blum, A., Konjevod, G., Ravi, R., Vempala, S.: Semi-definite relaxations for minimum bandwidth and other vertex-ordering problems. In: Proceedings of the 30th ACM Symposium on the Theory of Computing, pp. 95–100 (1997)
3. Byrne, K.P., Wolfe, K.H.: The Yeast Gene Order Browser: combining curated homology and syntenic context reveals gene fate in polyploid species. Genome Research 15, 1456–1461 (2005)
4. Cuthill, E., McKee, J.: Reducing the bandwidth of sparse symmetric matrices. In: Proceedings of the 24th National Conference of the ACM, pp. 157–172 (1969)
5. Felsenstein, J.: Inferring phylogenies. Sinauer Associates, Sunderland, MA (2004)
6. George, A.: Computer implementation of the finite element method, STAN-CS-71-208, Computer Science Dept., Stanford Univ., Stanford, CA (1971)
7. Hoberman, R., Sankoff, D., Durand, D.: The statistical analysis of spatially clustered genes under the maximum gap criterion. Journal of Computational Biology 12, 1081–1100 (2005)
8. Liu, J., Sherman, A.: Comparative analysis of the Cuthill-Mckee and the reverse Cuthill-Mckee ordering algorithms for sparse matrices. SIAM Journal of Numerical Analysis 13, 198–213 (1975)
9. Papadimitriou, C.H.: The NP-completeness of the bandwidth minimization problem. Computing 16, 263–270 (1976)
10. Saxe, J.: Dynamic-programming algorithms for recognizing small-band-width graphs in polynomial time. SIAM Journal of Algebraic and Discrete Methods 1, 363–369 (1980)

Physicochemical Correlation between Amino Acid Sites in Short Sequences under Selective Pressure

David Campo, Zoya Dimitrova, and Yuri Khudyakov

Molecular Epidemiology & Bioinformatics Laboratory, Division of Viral Hepatitis, Centers for Disease Control and Prevention. 1600 Clifton Rd, Atlanta, GA 30333
{fyv6,izd7,yek0}@cdc.gov

Abstract. The activities and properties of proteins are the result of interactions among their constitutive amino acids. In the course of natural selection, substitutions which tend to destabilize a particular structure may be compensated by other substitutions which confer stability to that structure. Patterns of coordinated substitutions were studied in two sets of selected peptides. The first is a set of 181 amino acid sequences that were selected *in vitro* to bind a MHC class I molecule (K^b). The second is a set of 114 sequences of the Hypervariable Region 1 of Hepatitis C virus, which, originating from infected patients, result from natural selection *in vivo*. The patterns of coordinated substitutions in both datasets showed many significant structural and functional links between pairs of positions and conservation of specific selected physicochemical properties.

Keywords: physicochemical properties, amino acid, covariation, selection.

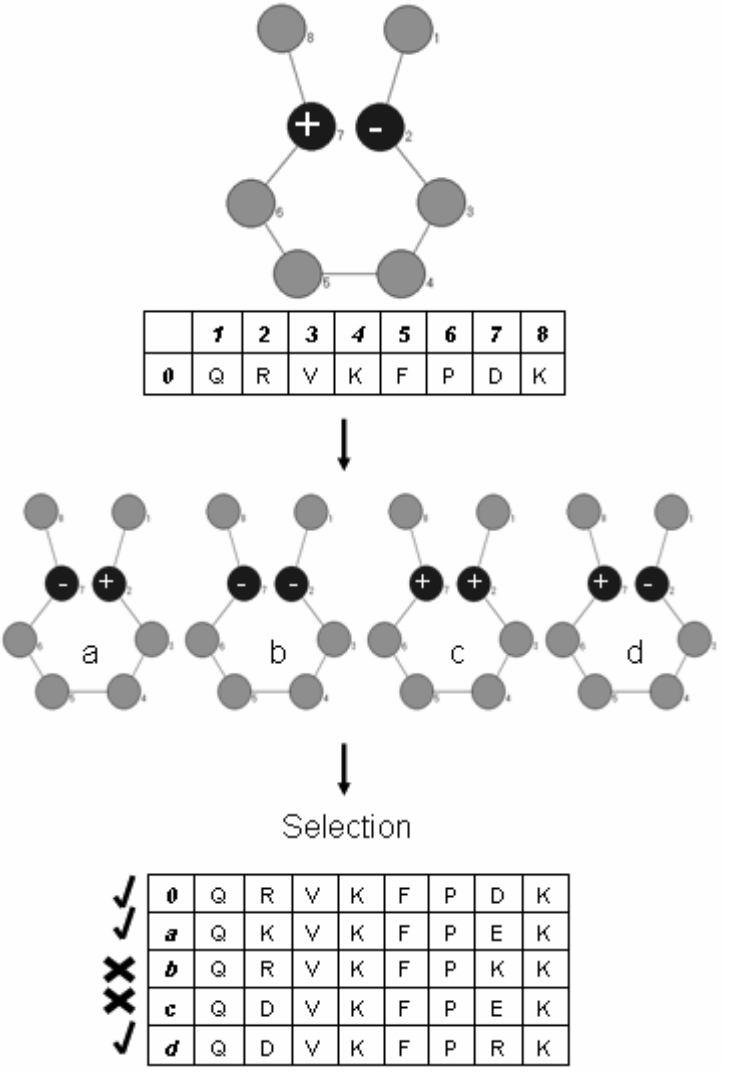
1 Introduction

Experimental and quantitative analyses of proteins often assume that the protein sites are independent, i.e., the presence of a residue at one site is independent of residues at other sites. However, the activities and properties of proteins are the result of interactions among their constitutive amino acids (aa) and, therefore, substitutions which tend to destabilize a particular structure and/or function are probably compensated by other substitutions that confer stability [1]. For example, if a salt bond were important, a substitution of the positively-charged residue with a neutral residue would need to be compensated by a nearby residue substituting from a neutral to a positive residue (Fig. 1). Similarly, a substitution involving a reduction of volume in the protein core might cause a destabilizing pocket which only one or a few adjacent residues would be capable of filling. Sites which are structurally or functionally linked will tend to evolve in a correlated fashion due to the compensation process [1].There is experimental evidence indicating that proteins contain pairs of covariant sites, identified both by analysis of families of natural proteins with known structures [2-7] and by site-directed mutagenesis whereby individual changes are introduced in proteins [8-10].

Independent mutations among functionally-linked sites would be disadvantageous, but simultaneous or sequential compensating mutations may allow the protein to retain function [11]. Furthermore, there are constraints on aa replacements that arise for functional reasons, such as aa bias at recognition sites related to DNA binding in

I. Măndoiu, R. Sunderraman, and A. Zelikovsky (Eds.): ISBRA 2008, LNBI 4983, pp. 146–158, 2008.

transcriptional regulators. Evolutionarily-related sequences should contain the vestiges of these effects in the form of covariant pairs of sites [12] and these interactions can be manifested in covariation between substitutions at pairs of alignment positions in a multiple sequence alignment. The analysis of covariation has been used in protein engineering [13], sequence-function correlations [14, 15], protein structure prediction [5, 12, 16-26] and in finding important motifs in viral proteins [27-30]. Recent analyses confirmed that highly coordinated sites are often functionally related and/or



	1	2	3	4	5	6	7	8
0	Q	R	V	K	F	P	D	K

0	Q	R	V	K	F	P	D	K
a	Q	K	V	K	F	P	E	K
b	Q	R	V	K	F	P	K	K
c	Q	D	V	K	F	P	E	K
d	Q	D	V	K	F	P	R	K

Fig. 1. Schematic representation of coordinated substitutions in a pair of aa sites forming a salt bond in a protein domain. Sequences that contain residues of the same charge at positions 2 and 7 are unstable (b and c) and are eliminated during natural selection. Sequences containing residues of different charges that are stable (o, a and d) can occur in a multiple sequence alignment.

spatially coupled, with coevolving positions being frequently located in regions critical for protein function, such as active sites and surfaces involved in molecular interactions with other proteins [14, 31-35].

In this paper, the patterns of coordinated substitutions were studied in two sets of selected peptides. The first is a set of 181 aa sequences that had been selected *in vitro* to bind a MHC class I molecule (K^b) [36, 37]. The second is a set of 114 sequences from Hypervariable Region 1 (HVR1) of Hepatitis C virus (HCV) generated *in vivo* from infected patients, which was used to understand the effects of natural selection on the pattern of coordinated substitutions. In both cases the process of selection over the structure and/or function of the peptide constrained the sequence variability and we found vestiges of these effects in the form of covariant pairs of sites.

2 Methods

2.1 Datasets

A previously published dataset of 310 peptides [36, 37] was used to investigate the effects of *in vitro* selection on the pattern of coordinated substitutions. This dataset gives the peptide aa sequence and corresponding binding to a MHC class I molecule (K^b) as a binary (yes/no) outcome. The complete dataset has 310 such observations (181 binders and 129 non-binders) and was originally obtained by random sampling from a large ($>10^7$) library of peptides, so there is no evolutionary history linking the peptides. The binding between two proteins generally involves short-range non-covalent interactions based on electrostatic charge, hydrogen bonding and van der Waals interactions. The specificity of the binding depends on the physicochemical properties of the constituent aa residues of both molecules and, therefore, the binding to the MHC class I molecule must select the conservation of some physicochemical properties in this subset of aa sequences. In this paper we wanted to know if the selection for this known function (binding) could be detected in the form of physicochemical correlation between aa sites.

HCV is a major cause of liver disease worldwide. The global prevalence of HCV infection is estimated to be 2.2%, representing 130 million people [38]. HCV causes chronic infection in 70-85% of infected adults [39]. There is no vaccine against HCV and current anti-viral therapy is relatively toxic, being effective in 50–60% of patients treated [40]. HCV is a single-stranded RNA virus of approximately 9400 nucleotides belonging to the *Flaviviridae* family [41]. The HVR1, located between positions 384 to 410 of the structural E2 protein, is the most intensively studied part of the HCV genome. However, the understanding of its function remains very limited [42] and it is not clear whether its high genetic heterogeneity is an immunological decoy or is related to a biologically relevant function [43]. Here, we studied the sequence variability of HVR1 in order to establish physicochemical correlations between aa sites, which could be due to the pressure of an unknown function that selected the conservation of physicochemical properties.

2.2 Sequences and Alignment

The previously published dataset of 310 peptides [36, 37] was obtained from the following web address: http://newfish.mbl.edu/Lab/Resources. All these peptides were

created in the same phage display library and had the same length so they were easily aligned. Two hundred and eight complete genome HCV sequences were obtained from "The Los Alamos HCV Sequence Database" [44] during early 2006. Of these 208 sequences, the following were excluded: recombinants, chimeras, patents, non-human hosts, a genotype other than 1b, and epidemiologically related sequences. This process left 114 different HCV 1b complete genome sequences, which were aligned using ClustalW [45]. The viral protein H77 (GenBank Accession Number NC_004102) was used as a reference sequence throughout this study.

2.3 Physicochemical Properties of aa

A study by Chelvanayagam et al. [31] found that the analysis of covariation involving different physicochemical characteristics improves the number of truly covariant pairs. However, there are many reported aa properties and the selection of the right ones presents a difficult choice. Interestingly, Atchley et al. [46] used multivariate statistical analyses on 494 aa properties [47] to produce a small set of highly interpretable numeric patterns of aa variability that can be used in a wide variety of analyses directed toward understanding the evolutionary, structural, and functional aspects of protein variability. This transformation summarizes the high level of redundancy in the original physicochemical attributes and produces much smaller, statistically independent, and well conditioned variables for subsequent statistical analysis [48]. The resultant factors are linear functions of the original data, fewer in number than the original, and reflect clusters of covarying traits that describe the underlying structure of the variables [46].

Factor analysis of the highly intercorrelated aa attributes resulted in five factors, a reduction in dimensionality of two orders of magnitude from the original 494 properties [46]. POLARF1 reflects polarity and simultaneous covariation in portion of exposed residues versus buried residues, non-bonded energy versus free energy, number of hydrogen bond donors, polarity versus non-polarity, and hydrophobicity versus hydrophilicity. HELIXF2 is a secondary structure factor. There is an inverse relationship of relative propensity for various aa in various secondary structural configurations, such as a coil, a turn, or a bend versus the frequency in an α-helix. SIZEF3 relates to molecular size or volume with high factor coefficients for bulkiness, residue volume, average volume of a buried residue, side chain volume, and molecular weight. CODONF4 reflects relative aa composition in various proteins and the number of codons for each aa. These attributes vary inversely with refractivity and heat capacity. CHARGEF5 refers to electrostatic charge with high coefficients on isoelectric point and net charge. Atchley et al. [48] showed how the transformation into one of the five multidimensional factors of physicochemical properties was useful in the analysis of Basic Helix-Loop-Helix proteins that bind DNA.

2.4 Multi-Response Permutation Procedure (MRPP)

MRPP is a non-parametric permutation test for testing the hypothesis of no difference between two or more groups of entities [49]. Permutation tests represent the ideal situations where one can derive the exact probabilities associated with a test statistic, rather than approximate values obtained from common probability distributions, such as t, F and X^2 [50]. In the majority of studies, the population distribution is unknown and assuming a normal distribution is inappropriate for many biological datasets,

which often are skewed, discontinuous, and multi-modal. The distance-functions that form the basis of the MRPP are used to detect differences in distributions, sensitive to both dispersion (variation) and shifts in central tendency (median) [51]. MRPP was used to test differences between the physicochemical properties of the two groups of MHC-binding peptides, which was performed by the program BLOSSOM [51].

2.5 Discriminant Analysis

Discriminant analysis is a statistical approach that defines the latent structure of between-groups covariation and determines the subset of attributes that best separate a set of a priori defined groups [52]. We used stepwise discriminant analysis (SWDA), as described by Atchley et al [48], to rank the 40 transformed variables of the MHC-binding peptides (5 physicochemical factors x 8 aa sites) in terms of their ability to discriminate between the binding and non-binding peptides. A step-up variable selection procedure begins with no variables in the model and then a variable is added that contributes most to discriminating power of the model, as measured by Wilks' lambda likelihood ratio criterion [52]. The procedure continues adding the next best discriminating variable until the p value of the F statistic was higher than 0.01. Then, we used Canonical variate analysis (CVA) to predict the membership of each peptide using the best discriminating variables of the SWDA model. A Leave-one-out cross validation procedure was employed, where each case is classified by the functions derived from all cases other than that case. All these procedures were performed with the program SPSS 15.0 [53].

2.6 Physicochemical Correlation between aa Sites

Bioinformatics methods for detecting correlated mutations consist of two main steps: (i) alignment of homologous sequences and (ii) identification of pairs of columns in the alignment in which there is a statistically significant tendency for mutations in one column to be accompanied by corresponding and usually different mutations in the other column [54]. In the present study, a modified version of a recent algorithm [55] was used to analyze pair-wise relationships between aa sites. The approach is based on estimation of the correlation coefficient between the values of a physicochemical parameter at a pair of positions of sequence alignment. When the correlation coefficient between two sites is negative, an increase in the value of a property at position i will make more likely a substitution at position j that will result in a decrease in the value of the property, which suggests a net value compensatory substitution. When the correlation coefficient is positive, it may be assumed that substitutions keep constant the difference between the property values of two residues. All statistical analyses, calculations and randomizations of this study were performed using MATLAB [56] unless stated otherwise.

If the sequences are effectively unrelated, then the pairs of positions with a significant covariation must have structural or functional links. Sequences are unrelated if the relationships by descent have been lost and there is no longer a significant phylogenetic signal or the sequences were obtained by in vitro selection (such as the dataset of MHC-binding peptides) [55]. There are three different sources of covariation in related biological sequences (such as the dataset of HCV sequences): (i) chance, (ii) common ancestry, and (iii) structural or functional constraints. Effectively discriminating among these underlying causes is a difficult task with many statistical

and computational difficulties [57]. There are many methods to test whether a correlation value reflects a significant association (possibly due to structural and functional constraints), or results from evolutionary history and stochastic events (background covariation) [14] but no single method has demonstrated general utility or achieved widespread acceptance [32]. We used the following four criteria to define the pairs of significantly correlated pairs of sites.

(i) Each one of the two sites has an entropy higher than 0.2370, which is 10% of the highest entropy found in the HCV polyprotein. Only 448 aa sites of the 3010 aa sites of the HCV polyprotein are above this entropy cutoff. This cut-off was chosen because prior modeling of protein coevolution showed that it is difficult to identify sites which are coevolving if they are highly conserved [32, 33].

(ii) A permutation procedure was performed, whereby the aa at each site in the sequence alignment was vertically shuffled. Ten thousand random alignments were created this way, simulating the distribution of correlation values under the null hypothesis that substitutions of aa at two sites are statistically independent. For each physicochemical factor, a pair of sites was considered significantly correlated if its correlation value in the observed dataset was higher than the correlation value for those two sites in any of the random datasets ($p = 0.0001$). We addressed the multiple comparisons problem with the False Discovery Rate approach, which controls the expected proportion of false positive results [58]. The False Discovery Rate in our study has a q-value of 0.00035 for the dataset of MHC-binding peptides and 0.00506 for the dataset of viral sequences.

(iii) Related sequences (such as the dataset of HCV sequences) are part of a hierarchically structured phylogeny and, therefore, for statistical purposes, cannot be regarded as being drawn independently from the same distribution. We used the data weighting approach based on Felsenstein's method [59] in the calculation of the correlation values, which is based on the assumption that the lower the time of divergence of two sequences from their common ancestor, the higher is the covariation between these two sequences. The one-dimensional weights were calculated using a distance matrix among sequences built using the synonymous sites of the full HCV genome.

(iv) We used a modified version of the method of Martin et al [32] and Gloor et al [33]. This method makes the assumption that each position in a multiple sequence alignment is affected equally by background correlation, and that the majority of positions in the alignment covary only because of common ancestry. On the basis of these assumptions, each alignment is used as its own null model for the identification of covarying positions. A critical correlation threshold was calculated using the value of the Student's distribution at a given significance level ($p = 0.001$) with a sample size of 114 sequences, following Afonnikov et al. [23].

3 Results

3.1 Dataset of MHC-Binding Peptides

There are two structurally distinct groups of peptide sequences in the dataset: those which bind to a MHC class I Kb molecule and those which do not. The dataset was

transformed to the five physicochemical factors and it was found that these two groups occupy a significantly different region of the physicochemical space ($p = 0.0001$; MRPP). Now that we know that the two classes of peptides have significant physicochemical differences, it is important to understand the causes of these differences. We used SWDA to rank the 40 transformed variables of the MHC-binding peptides in terms of their ability to discriminate between binders and non-binders. The results indicate that some positions and factors contribute much more strongly than others in separating the data (Table 1), positions 5 and 8 being the most important, especially regarding the physicochemical properties summarized by POLARF1 and HELIXF2. This model includes 8 variables that account for 70.28% of the variability, and allow the correct classification of 91.3% of the peptides (90.3 of cross-validated cases are correctly classified) (Table 2).

Table 1. Discriminant analysis (SWDA) of the MHC-binding peptides

Step	Variable	Residual Variance	Significance of F
1	SIZEF3_P5	0.6310	2.5751E-06
2	HELIXF2_P8	0.4851	1.4468E-07
3	POLARF1_P8	0.4229	1.0112E-06
4	POLARF1_P5	0.3722	2.3665E-15
5	CODONF4_P5	0.3298	1.6459E-10
6	SIZEF2_P1	0.3117	2.2650E-04
7	SIZEF2_P3	0.3048	3.5888E-03
8	POLARF1_P2	0.2973	6.6078E-03

Table 2. Classification results of the CVA 8-variable model

		Predicted binders	Predicted non-binders
	Observed binders	92.82%	7.18%
Original	**Observed non-binders**	10.85%	89.15%
	Observed non-binders	92.27%	7.73%
Cross-validated	**Observed non-binders**	12.40%	87.60%

What physicochemical characteristics are the sequences of binders keeping constant? Which pairs of sites are highly associated in order to conserve affinity? There are eight pairs of positions with significant correlations (links) in the set of binders (Table 3), but none in the set of non-binders (Fig. 2). Almost all positions in the MHC-binding dataset have one or more links, except position 8. However, position 8 has a link to position 7 with a lower significance ($p = 0.0012$). Position 3 has the highest number of links, followed by positions 1 and 5. The correlated positions in binding sequences keep constant factors POLARF1, SIZEF3 and CHARGEF5 of the peptide. These results suggest that our simple covariation analysis is useful for finding pairs of sites that are crucial to keeping the structural conformation of peptides.

Table 3. Significant physicochemical correlations at one or more factors (p = 0.0001) in binders

i	j	POLARF1	HELIXF2	SIZEF3	CODONF4	CHARGEF5
1	3			-0.2823		
1	5	0.2701				
1	6			-0.3889		-0.3520
2	3	0.4408		0.3357		0.2821
2	5			-0.3087	-0.3617	-0.3019
3	5	0.3275	0.4030			
3	7	-0.2822				
4	6				0.3119	

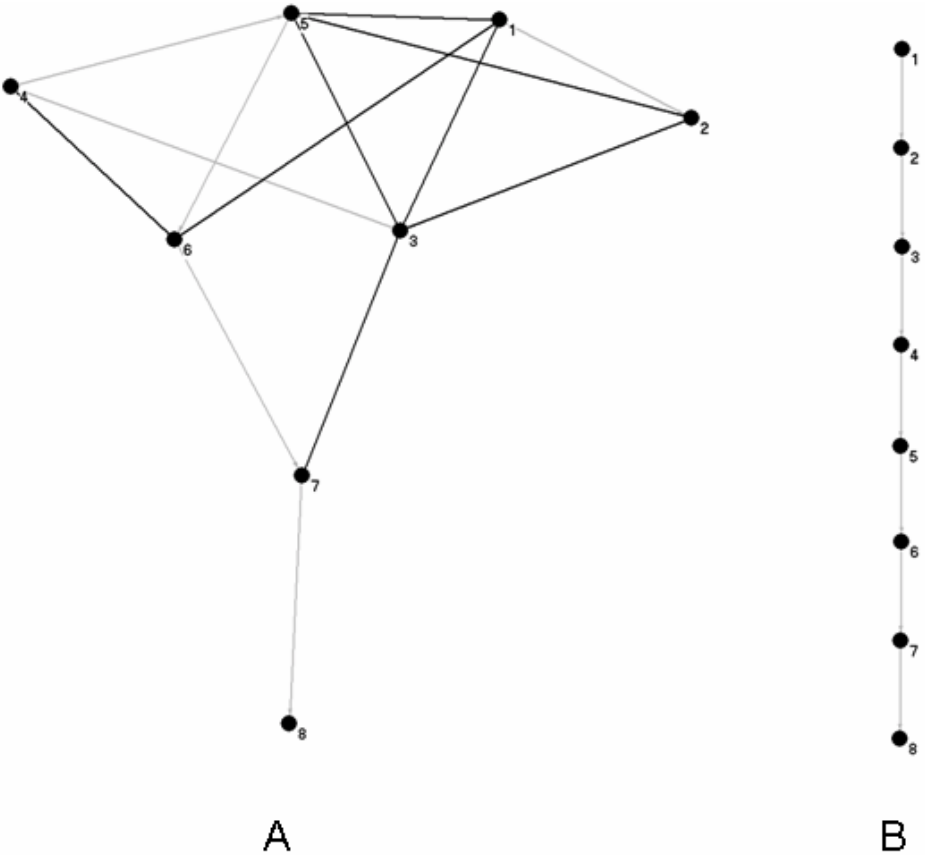


Fig. 2. Graph of the relationships between sites in the binders (A) and non-binders (B). Contiguous sites in the sequence are connected *(grey lines)* and sites with a significant physicochemical correlations at one or more factors (p = 0.0001) are also connected *(black lines)*. There are eight pairs of positions with significant correlations in the set of binders, but none in the set of non-binders.

3.2 HVR1

We found five links involving eight different sites in the middle of HVR1 (Table 4 and Fig. 3). The site with the highest number of links is 402 (3 links), suggesting an important role in keeping HVR1 structure and/or function. Changes in the HVR1 are correlated in a way that keeps constant the POLARF1 and SIZEF3 of the segment. The results suggest that there are three physicochemical traits or conditions that have

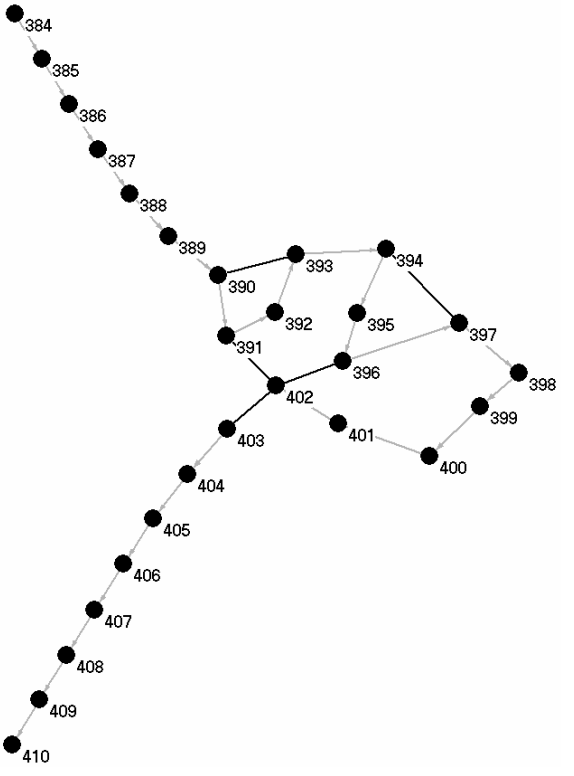


Fig. 3. Graph illustrating relationships between sites in the HVR1. Contiguous sites in the sequence are connected *(grey lines)* and sites with a significant physicochemical correlation at one or more factors (p = 0.0001) are also connected *(black lines)*.

Table 4. Significant physicochemical correlations at one or more factors (p = 0.0001) in HVR1

i	j	POLARF1	HELIXF2	SIZEF3	CODONF4	CHARGEF5
390	393			-0.3063		
391	402	0.4090				
394	397	-0.3759				
396	402	0.3522				
402	403				-0.5505	

been selected in the HVR1: the first in a cluster of sites (391, 396, 402 and 403) related to POLARF1 where site 402 seems critical; the second (390 and 393) related to SIZEF3, and the third (394 and 397) related to POLARF1.

4 Discussion

Knowledge about the determinants of binding to MHC molecules is very useful for the development of predictive tools that help choose peptides for employment in immunological therapies or inclusion in vaccines intended to elicit T-cell cytotoxic activity. It has been shown that some aa residues occur at a high frequency at specific positions in the peptide, termed anchor positions [36, 37]. For the MHC class I molecule (K^b) the anchor positions are 3, 5 and 8, with preferences for aa tyrosine or phenylalanine in positions 3 or 5 and a hydrophobic aa in position 8. However, binding is known to be influenced by both the presence of secondary anchor positions and interactions between aa within the peptide [37]. Interestingly, we found that positions 3, 5 and 8 are the best positions in discriminating between binders and non-binders, in agreement with their role as anchor positions. We also found that there is a high level of covariation between all positions in MHC binding peptides but none in the set of non-binding peptides. This observation clearly suggests that the ability to bind to MHC creates constraints on the sequence variability of the binding peptides, where changes at some positions are coordinated with changes at other positions in order to maintain binding capacity. The covariation level of position 8 is very low, suggesting that its contribution to MHC binding is more independent of the other sites, even though it is an anchor position and is the most important position discriminating between binders and non-binders. The aa distribution of position 8 is very different in the binding and non-binding dataset, with significant differences in the average POLARF1 ($p = 0.0001$; MRPP) and HELIXF2 ($p = 0.0001$; MRPP) of the two groups. These results suggest that there are two physicochemical traits or conditions affecting binding in these 8-mer peptides: the first is related to positions 1-7 (conserving POLARF1, SIZEF3 and CHARGEF5) and the second is related with position 8 (POLARF1 and HELIXF2), which is less dependent of the other positions but very important to define the ability to bind.

We also studied the sequence variability of HVR1 in order to establish if there is a mechanism underlying the selection of this subset of aa sequences. The results suggest that the HVR1 segment has to keep a specific structure and/or function and that natural selection left a mark in the sequence variability in the form of coordinated substitutions. The high number of coordinated substitutions and their contribution to the maintenance of some physicochemical values provide additional proof to the conservation of conformational motifs in the HVR1, for which there is previous experimental evidence [43]. This conservation is consistent with strong selective constraints previously found on HVR1 heterogeneity [42, 60, 61], suggesting that this segment has an important function in virus replication, rather than merely being a variable region of the genome that acts as an antigenic decoy [42, 60]. The results suggest that the physicochemical properties POLARF1 and SIZEF3 have been selected in the HVR1. The high number of coordinated substitutions and their contribution to the integrity of these physicochemical properties provide an additional proof to conservation of conformational motifs in the HVR1. Covariation analyses can be important in identifying sites that may change the phenotype of a protein, and they could be used as a tentative map for researchers to define functional domains in the protein through

mutational analysis. For instance, covariant sites could be used as a guide for rational selection of sets of sequences for inclusion in a mixture of peptides for vaccine design. Therefore, by selecting sequences which include pairs of aa that are highly predictive of each other, important classes of sequences that are structurally or functionally related may be identified. Thus inclusion of peptides with highly covariant aa may be a useful strategy for designing broadly-reactive vaccines [28].

The detection of coordinated substitutions among separate aa sites is fundamental to understanding protein structure and evolution. The process of selection (whether natural or *in vitro*) creates profound constraints on the sequence variability, keeping constant the structure or function of the protein. We found the consequences of these constraints in the form of covariant pairs of sites and the conservation of specific physicochemical properties.

Acknowledgments. The authors are grateful to Dr. Chong-Gee Teo (Centers for Disease Control and Prevention, Atlanta, GA) and Dr. Robert Mitchell (La Trobe University, Melbourne, Australia) for valuable comments at different stages of the project.

References

1. Pollock, D., Taylor, W.: Effectiveness of correlation analysis in identifying protein residues. Protein Eng. 10(6), 647–657 (1997)
2. Chothia, C., Lesk, A.: Evolution of proteins formed by beta-sheets. I. Plastocyanin and azurin. J. Mol. Biol. 160(2), 309–323 (1982)
3. Lesk, A., C., C.: Evolution of proteins formed by beta-sheets. II. The core of the immunoglobulin domains. J. Mol. Biol. 160(2), 325–342 (1982)
4. Oosawa, K., Simon, M.: Analysis of mutations in the transmembrane region of the aspartate chemoreceptor in Escherichia coli. Proc. Natl. Acad. Sci. USA 83(18), 6930–6934 (1986)
5. Altschuh, D., et al.: Coordinated amino acid changes in homologous protein families. Protein Eng. 2(3), 193–199 (1988)
6. Bordo, D., Argos, P.: Evolution of protein cores. Constraints in point mutations as observed in globin tertiary structure. J. Mol. Biol. 211(4), 975–988 (1990)
7. Mateu, M., Fersht, A.: Mutually compensatory mutations during evolution of the tetramerization domain of tumor supressor p53 lead to impaired hetero-oligomerization. Proc. Natl. Acad. Sci. USA 96, 3595–3599 (1999)
8. Lim, W., Sauer, R.: Alternative packing arrangements in the hydrophobic core of lambda repressor. Nature 339(6219), 31–36 (1989)
9. Lim, W., Farruggio, D., Sauer, R.: Structural and energetic consequences of disruptive mutations in a protein core. Biochemistry 31(17), 4324–4333 (1992)
10. Baldwin, E., et al.: The role of backbone flexibility in the accommodation of variants that repack the core of T4 lysozyme. Science 262(5140), 1715–1718 (1993)
11. Govindarajan, S., et al.: Systematic variation of Amino acid substitutions for stringent assesment of pairwise covariation. J. Mol. Biol. 328, 1061–1069 (2003)
12. Clarke, N.: Covariation of residues in the homeodomain sequence family. Protein Sci. 4(11), 2269–2278 (1995)
13. Voigt, C., et al.: Computational method to reduce the search space for directed protein evolution. In: Proc. Natl. Acad. Sci. USA, vol. 98, pp. 3778–3783 (2001)
14. Atchley, W., et al.: Correlations among amino acid sites in bHLH protein domains: an information theoretic analysis. Mol. Biol. Evol. 17(1), 164–178 (2000)

15. Fukami-Kobayashi, K., Schreiber, D., Benner, S.: Detecting compensatory covariation signals in protein evolution using reconstructed ancestral sequences. J. Mol. Biol. 319, 729–743 (2002)
16. Göbel, U., et al.: Correlated mutations and residue contacts in proteins. Proteins 18(4), 309–317 (1994)
17. Neher, E.: How frequent are correlated changes in families of protein sequences? Proc Natl Acad Sci USA 91(1), 98–102 (1994)
18. Shindyalov, I., Kolchanov, N., Sander, C.: Can three dimensional contacts in protein structures be predicted by analysis of correlated mutations? Protein Eng. 7, 349–358 (1994)
19. Taylor, W., Hatrick, K.: Compensating changes in protein multiple sequence alignments. Protein Eng. 7(3), 341–348 (1994)
20. Benner, S., et al.: Bona fide predictions of protein secondary structure using transparent analyses of multiple sequence alignments. Chem. Rev. 97, 2725–2844 (1997)
21. Nagl, S., Freeman, J., Smith, T.: Evolutionary constraint networks in ligand-binding domains: an information-theoretic approach. Pac. Symp. Biocomput, 90–101 (1999)
22. Larson, S., Di Nardo, A., Davidson, A.: Analysis of covariation in an SH3 domain sequence alignment: applications in tertiary contact prediction and the design of compensating hydrophobic core substitutions. J. Mol. Biol. 303(3), 433–446 (2000)
23. Afonnikov, D., Oshchepkov, D., Kolchanov, N.: Detection of conserved physico-chemical characteristics of proteins by analyzing clusters of positions with co-ordinated substitutions. Bioinformatics 17(11), 1035–1046 (2001)
24. Nemoto, W., et al.: Detection of pairwise residue proximity by covariation analysis for 3D-structure prediction of G-protein-coupled receptors. Protein. J. 23(6), 427–435 (2004)
25. Wang, L.: Covariation analysis of local amino acid sequences in recurrent protein local structures. J. Bioinform. Comput. Biol. 3(6), 1391–1409 (2005)
26. Shackelford, G., Karplus, K.: Contact prediction using mutual information and neural nets. Proteins 69(suppl. 8), 159–164 (2007)
27. Altschuh, D., et al.: Correlation of co-ordinated amino acid substitutions with function in viruses related to tobacco mosaic virus. J. Mol. Biol. 193(4), 693–707 (1987)
28. Korber, B., et al.: Covariation of mutations in the V3 loop of human immunodeficiency virus type 1 envelope protein: an information theoretic analysis. Proc. Natl. Acad. Sci. USA 90(15), 7176–7180 (1993)
29. Gilbert, P., Novitsky, V., Essex, M.: Covariability of selected amino acid positions for HIV type 1 subtypes C and B. AIDS Res. Hum. Retroviruses 21(12), 1016–1030 (2005)
30. Kolli, M., Lastere, S., Schiffer, C.: Co-evolution of nelfinavir-resistant HIV-1 protease and the p1-p6 substrate. Virology 347(2), 405–409 (2006)
31. Chelvanayagam, G., et al.: An analysis of simultaneous variation in protein structures. Protein Eng. 10(4), 307–316 (1997)
32. Martin, L., et al.: Using information theory to search for co-evolving residues in proteins. Bioinformatics 21(22), 4116–4124 (2005)
33. Gloor, G., et al.: Mutual information in protein multiple sequence alignments reveals two classes of coevolving positions. Biochemistry 44(19), 156–165 (2005)
34. Poon, A., Chao, L.: The rate of compensatory mutation in the DNA bacteriophage phiX174. Genetics 170(3), 989–999 (2005)
35. Yeang, C., Haussler, D.: Detecting coevolution in and among protein domains. PLoS Comput Biol. 3(11), e211 (2007)
36. Milik, M.S., Brunmark, D., Yuan, A., Vitiello, L., Jackson, A., Peterson, M., Skolnick, P., Glass, J.: Application of an artificial neural network to predict specific class I MHC binding peptide sequences. Nat. Biotechnol. 16(8), 753–756 (1998)

37. Segal, M., Cummings, M., Hubbard, A.: Relating amino acid sequence to phenotype: analysis of peptide-binding data. Biometrics 57(2), 632–642 (2001)
38. Alter, M.: Epidemiology of hepatitis C virus infection. World J. Gastroenterol. 13(17), 2436–2441 (2007)
39. Alberti, A., Chemello, L., Benvegnu, L.: Natural History Of Hepatitis C. J. Hepatol. 31(supp. 1), 17–24 (1999)
40. Bowen, D., Walker, C.: Adaptive immune responses in acute and chronic hepatitis C virus infection. Nature 436, 946–952 (2005)
41. Choo, Q., et al.: Isolation Of A Cdna Clone Derived From A Bloodborne Non-A, Non-B Viral Hepatitis Genome. Science 244, 359–362 (1989)
42. Smith, D.: Evolution of the hypervariable region of hepatitis C virus. J. Viral Hepat 6(suppl. 1), 41–46 (1999)
43. Mondelli, M., et al.: Hypervariable region 1 of hepatitis C virus: immunological decoy or biologically relevant domain? Antiviral Res. 52(2), 153–159 (2001)
44. Kuiken, C., et al.: The Los Alamos hepatitis C sequence database. Bioinformatics 21(3), 379–384 (2005)
45. Thompson, J., Higgins, D., Gibson, T.: CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. Nucleic. Acids. Res. 22(22), 4673–4680 (1994)
46. Atchley, W., et al.: Solving the protein sequence metric problem. Proc. Natl. Acad. Sci. USA 102(18), 6395–6400 (2005)
47. Kawashima, S., Kanehisa, M.: AAindex: amino acid index database. Nucleic. Acids. Res. 28, 374 (2000)
48. Atchley, W., Zhao, J.: Molecular architecture of the DNA-binding region and its relationship to classification of basic helix-loop-helix proteins. Mol. Biol. Evol. 24(1), 192–202 (2007)
49. McCune, B., Grace, J.: Analysis of ecological communities, MjM Software Design, Gleneden Beach (2002)
50. Cai, L.: Multi-response Permutation Procedure as An Alternative to the Analysis of Variance: An SPSS Implementation. Department of Psychology, University of North Carolina (2004)
51. Cade, B., Richards, J.: User Manual For BLOSSOM Statistical Software. Midcontinent Ecological Science Center US Geological Survey Fort Collins, Colorado (2001)
52. Johnson, R., Wichern, D.: Applied multivariate statistical analysis. Prentice Hall, Upper Saddle River, NJ (2002)
53. SPSS 15.0 for windows, SPSS Inc, Chicago IL (2006)
54. Noivirt, O., Eisenstein, M., Horovitz, A.: Detection and reduction of evolutionary noise in correlated mutation analysis. Protein Eng. 18(5), 247–253 (2005)
55. Afonnikov, D., Kolchanov, N.: CRASP: a program for analysis of coordinated substitutions in multiple alignments of protein sequences. Nucleic. Acids. Res. 32, W64–W68 (2004)
56. MathWorks, T.: MATLAB, Natick, MA (2007)
57. Wollenberg, K., Atchley, W.: Separation of phylogenetic and functional associations in biological sequences by using the parametric bootstrap. Proc. Natl. Acad. Sci. USA 97(7), 3288–3291 (2000)
58. Benjamini, Y., Hochberg, Y.: Controlling the false discovery rate: a practical and powerful approach to multiple testing. Journal of the Royal statistical Society, Series B 57(1), 289–300 (1995)
59. Felsenstein, J.: Phylogenies and the comparative method. Am. Nat. 125, 1–15 (1985)
60. McAllister, J., et al.: Long-term evolution of the hypervariable region of hepatitis C virus in a common-source-infected cohort. J. Virol. 72(6), 4893–4905 (1998)
61. Sheridan, I., et al.: High-resolution phylogenetic analysis of hepatitis C virus adaptation and its relationship to disease progression. J. Virol 78(7), 3447–3454 (2004)

HCV Quasispecies Assembly Using Network Flows

Kelly Westbrooks[1,⋆], Irina Astrovskaya[1,⋆], David Campo[2], Yury Khudyakov[2], Piotr Berman[3], and Alex Zelikovsky[1]

[1] Department of Computer Science, Georgia State University, Atlanta, GA 30303
{csckew,iraa,alexz}@cs.gsu.edu
[2] Centers for Disease Control and Prevention, Atlanta, GA 30333
{fyv6,yek0}@cdc.gov
[3] Department of Computer Science and Engineering, Pennsylvania State University
University Park, PA 16802
berman@cse.psu.edu

Abstract. Understanding how the genomes of viruses mutate and evolve within infected individuals is critically important in epidemiology. By exploiting knowledge of the forces that guide viral microevolution, researchers can design drugs and treatments that are effective against newly evolved strains. Therefore, it is critical to develop a method for typing the genomes of all of the variants of a virus (quasispecies) inside an infected individual cell.

In this paper, we focus on sequence assembly of Hepatitis C Virus (HCV) based on 454 Lifesciences system that produces around 250K reads each 100-400 base long. We introduce several formulations of the quasispecies assembly problem and a measure of the assembly quality. We also propose a novel scalable assembling method for quasispecies based on a novel network flow formulation. Finally, we report the results of assembling 44 quasispecies from the 1700 bp long E1E2 region of HCV.

1 Introduction

Many viruses found in nature encode their genomes in RNA rather than DNA. While the problem of sequencing an organism's DNA is well-studied, sequencing RNA viruses presents its own unique set of challenges. Perhaps the biggest challenge associated with sequencing RNA viruses is that they lack DNA polymerases and are unable to repair mistakes in their sequences as they reproduce. Over the course of infection, the mistakes made in replication are passed down to descendants, producing a family of related variants of the original viral genome referred as a *quasispecies*.

The allele frequencies across all of the quasispecies in an infected individual may drift significantly. Among all of the new quasispecies produced, some may be more virulent than others. Thus, it is of epidemiological interest to identify common characteristics of virulent quasispecies to aid in the design of effective drugs and treatments for the disease that the virus causes. This paper is devoted to the problem of sequencing of all quasispecies inside a patient based on 454 Lifesciences system.

454 Lifesciences system is one promising technology that may prove useful for sequencing quasispecies. It is a massively-parallel pyrosequencing system developed by

⋆ Partially supported by GSU Molecular Basis of Disease Fellowship.

I. Măndoiu, R. Sunderraman, and A. Zelikovsky (Eds.): ISBRA 2008, LNBI 4983, pp. 159–170, 2008.

biotechnology firm 454 Lifesciences for DNA sequencing. Briefly, the system fragments the source genetic material to be sequenced into pieces approximately 100 bp long called *reads*. Each read is sequenced and the original genome is reconstructed via software. Since this system was originally designed to sequence genetic material from a single organism, the software assembles all of the reads to a single genome. In order to use it for sequencing quasispecies, new software must be created that can also correctly distribute reads between multiple quasispecies.

Informally, the Quasispecies Assembly problem can be stated as follows: Given a set of reads taken from a single specimen, determine how many quasispecies are present and what are their sequences.

Quasispecies Assembly is related to several well-known problems: DNA fragment assembly (see e.g., [2,5,6]), haplotype assembly [3], population phasing (see e.g., [7]) and DNA finding in a mixed environment (see e.g., [15]) . Indeed, the fragments (reads) should be assembled into a long genome sequence although it becomes a lesser challenge since consensus genome sequence is already available. In [2], a network flow-based approach is presented which bears similarity to the approach adopted in this article. A plausible reduction genome sequencing is as follows: place all quasispecies genomes back-to-back in a long sequence and treat common segments as repeats. Quasispecies Assembly is very close to the haplotype assembly problem where fragments are given from two different haplotypes of the same diploid organism and the goal is to correctly attribute segments to one of these two haplotypes. Unfortunately, the proposed solution methods are not scalable with respect to the number of haplotypes per individual and this is critical since in a specimen there are hundreds or even thousands of different quasispecies. Therefore, one can find similarity with the population phasing problem where multiple diplotypes (mixtures of two haplotypes) are given and it is required to identify underlying common haplotypes and their frequencies. Finally, it can also be viewed as variant of the newly-arisen problem of finding and distinguishing all the different species in a given DNA sample – but in our case, the complicating factor is that the sequencing of different quasispecies are very similar to each other.

Our contributions are the following:

- Several optimization formulations for Quasispecies Assembly and its different versions.
- Estimation of the probability that two overlapping read belong to the same quasispecies.
- A network flow based method for solving the quasispecies assembly problems.
- An efficient and scalable implementation of the proposed network flow methods.
- Application of the network flow method to the set of simulated reads drawn from 44 quasispecies in the E1E2 region of Hepatitis C Virus.

In the next section we give several optimization formulations for Quasispecies Assembly. Then we will construct proposed data structure incorporating information about given reads and the consensus genome. Section 4 will propose solutions for Quasispecies Assemble problems based on reductions to network flows. Finally, Section 5 will describe validation of network flow approaches on E1E2 region of HCV.

2 Quasispecies Assembly Models and Optimization Formulations

The ultimate goal of Quasispecies Assembly is to correctly reconstruct genomes of all quasispecies in a given sample. Since multiple quasispecies have indistinguishable common segments that are significantly longer than a read, one cannot guarantee to find even the exact number of quasispecies. Although only cross-validation of proposed techniques can really tell if their quality are of practical interest, it is important to formulate models and corresponding optimization objectives that do not simply rely on cross-validation.

We will start with the formal description of the input and output for Quasispecies Assembly. The output of 454 Lifescience system consists of $N \approx 250K$ reads, each read r is a sequence of l nucleotides (l may be about 100 or even 400). The rate of typing errors is claimed to be 0.04% (see [9]). Also we may usually rely on existence of a known consensus genome of all quasispecies which is in case of HCV has length $L = 9600$ bp. Each reconstructed quasispecies should be covered by given reads and be close to the consensus genome sequence H.

We first consider the simplest parsimonious model for Quasispecies Assembly. The corresponding optimization formulation is as follows.

Most Parsimonious Quasispecies Assembly. Given a set of reads R and a consensus genome sequence H, find the minimum size set of quasispecies Q covering all reads from R, i.e., such that each read $r \in R$ is contained in at least one $q \in Q$.

Although the parsimonious model is worth considering, it is usually oversimplified. Indeed, it usually predicts less than observed number of quasispecies and cannot distinguish between numerous different equally good (from parsimonious point of view) solutions. In order to break ties, we introduce penalties over read overlaps. The penalty $cost(r, r')$ over an overlap between reads $r, r' \in R$ should reflect how unsure we are that these two reads came from the same quasispecies. We set $cost(q)$ to be the sum of costs of constituting overlaps. For example, cost can be inversely proportional to the probability that such overlap occurs. Then the overall probability of the quasispecies q is the product of costs of consecutive overlapping pairs of reads which can be transformed to the sum by replacing costs with their logarithms. In the next section we will suggest several different cost functions.

Minimum Cost Parsimonious Quasispecies Assembly. Given a set of reads R with costs on read overlaps and a consensus genome sequence H, find the most parsimonious set of quasispecies Q that have the total minimum cost.

We may also trade the number of quasispecies for smaller cost or just completely disregard minimization of the number of quasispecies.

Minimum Cost Quasispecies Assembly. Given a set of reads R with costs on read overlaps and a consensus genome sequence H, find the set of quasispecies Q covering all reads that have the total minimum cost.

Besides accurately assembling all quasispecies as a set, it is also important to assemble certain frequent individual quasispecies. There is an evidence that the frequency

distribution of quasispecies in a single cell is usually not uniform. The most frequent quasispecies may contribute the major part of reads and may also contribute the most to virus resiliency. Although frequently repeated reads may come from the most frequent quasispecies, the alternative explanation would be that multiple different quasispecies have the same common segment. Therefore, we again rely on estimated probability for overlapping reads. This results in the following problem formulation.

The Most Frequent Quasispecies Assembly. Given a set of reads R with costs on read overlaps and a consensus genome sequence H, find a single quasispecies q with the minimum total cost.

3 The Read Graph

In this section we propose the method of incorporating the input information about reads and genome consensus sequence into a single data structure to which we apply network flow methods for solving quasispecies assembly problems.

We will first describe how to align reads to the consensus genome and distinguish single nucleotide polymorphisms (SNP) from typing errors. For every possible starting position, we align both the read and its reverse compliment to the consensus sequence and count the number of mismatches. The "true" starting position has the fewest number of mismatches. In our experiments we have never encountered read misalignments which can be explained by a lack of sizable repeats in the RNA viral genomes and the low typing error rate (0.04% [9]). We can distinguish typing errors from infrequent SNPs if we have at least double coverage of each quasispecies – indeed, the probability of the same typing error occurring twice is insignificantly small.

Formally, each read r is supplied with its beginning and ending positions in the consensus sequence b_r and e_r, respectively. The *read graph* $G = (V, E, cost)$ has the set of vertices V each representing a read and the set of directed edges E, where each edge (u, v) connects two reads u and v if their alignments overlap (i.e., $b_u \leq b_v \leq e_u \leq e_v$) and if they coincide with each other across the overlapping region.

Obviously, some edges correspond to *true* overlaps of pairs of reads coming from the same quasispecies while other correspond to false overlaps that occur between reads of similar but different quasispecies. The cost function of an edge (u, v) reflects how unsure we are that u and v correspond to a true overlap.

In the next subsection we describe how we reduce the size of the read graph without losing any information. Our reduction is based on an efficient algorithm for minimum transitive reduction. In Subsection 3.2 we estimate the probability for an edge in the transitively reduced read graph to correspond to a true overlap.

3.1 Transitive Reduction of the Read Graph

In general, the read graph G may be very dense since it contains edges connecting non-consecutive reads (see Figure 3.1). If there are three reads u, v and w such that $(u, v), (v, w) \in E$ and u overlaps with w (i.e., $b_w < e_u$), then $(u, w) \in E$. The path $u - v - w$ is called *closed* since there is a single edge (u, w) connecting the beginning

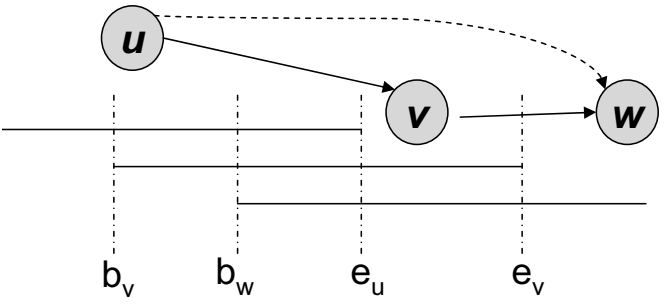


Fig. 1. The edge (u, w) is logically implied by the edges (u, v) and (v, w). Indeed, the segment $[b_w, e_w]$ is the same in the reads u and v since $(u, v) \in E$ and $[b_w, e_w]$ is the same in the reads v and w since $(v, w) \in E$, therefore, it is the same for u and w.

with the end. The edge (u, w) is logically implied by the other edges and we can safely remove it without losing any information.

Thus, we wish to remove maximum possible number of edges, or, in other words, obtain the minimum transitive reduction $G' = (V, E')$ of the graph G. The *transitive reduction* is a subgraph of G in which if a vertex v is reachable from u, then it should be reachable in G'. In general, finding minimum transitive reduction is NP-complete but since G is is a directed acyclic graph, it can be found efficiently [10]. Besides lacking directed cycles, the read graph G is also *partially transitively closed*, i.e., all subpaths of closed paths are closed.

Input: Partially transitively closed directed acyclic graph $G = (V, E)$
Output: Minimum transitive reduction of G

1. Topologically sort vertices of G
2. For each vertex $u \in V$ in topological order do
3. Sort all outgoing edges from u according to left end: $v_1, \ldots, v_k$
4. Thread set $T \leftarrow \emptyset$
5. For $i = 1, \ldots, k$ do
6. For each $x \in T$ do
   If edge $(x, v_i) \in E$, then $E \leftarrow E - (u, v_i)$, $T \leftarrow T - x$, break
   $T \leftarrow T \cup v_i$
7. Output G

Fig. 2. Minimum transitive reduction for partially transitively closed directed acyclic graph

Claim. A read graph G is partially transitively closed.

Proof. Toward contradiction assume that there exists a closed $u - v$-path P without chords. Let (w, v) be the last edge of P, we will show that there exists the edge (u, w). Indeed, existence of $u - w$-path and (w, v)-edge implies that $b_u \leq b_w \leq b_v \leq e_u$ and,

therefore, u and w overlap. Since there exists a $u - w$-path, u and w do not disagree. These two facts imply there should be an edge u and v.

The following algorithm for finding minimum transitive reduction (see Figure 2) is more efficient than for general directed acyclic graphs since it relies on G being partially transitively closed. The runtime is $O(\delta|E|)$, where δ is the maximum number of quasispecies containing the same read and $|E|$ is the number of edges in G. This is significantly faster than $O(|V|^2)$ for arbitrary directed acyclic graphs.

From now on, we assume that the read graph G is transitively reduced. Obviously, an arbitrary quasispecies corresponds to unextendable path of G, although not every unextendable path corresponds to a quasispecies.

3.2 Estimating Probability of a True Overlap

We first give intuition behind estimation of the true overlap probability and then present results of the formal analysis of the uniform and non-uniform quasispecies distributions.

Intuitively, given a choice, one would trust a larger read overlap more than a smaller read overlap. That makes a lot of sense in the standard sequencing when the consensus genome is unknown. The entire de Bruijn graph approach relies only on sufficiently long overlaps (see [4]). Indeed, it is quite improbable that a long overlap happens by chance – only repeats may result in false long read overlaps. But Quasispecies Assembly exactly the case with long and frequent repeats – many segments can be repeated in very many quasispecies.

Only multiple coverage may give a clue for deciding which overlaps are probably true. If there are two reads u and v adjacent in the transitively reduced read graph, then we may try to measure our surprise with the fact that $(u, v) \in E$ by the length of the "overhang" $\Delta = |b_v - b_u|$. Indeed, assuming that (u, v) represents a true overlap in a quasispecies q, why there is no other read w that is taken from q and which is between u and v? If Δ is large, then there great chance that the overlap is false.

Formally, let us consider a simplified model where every read has the same length and that each quasispecies has the same frequency.

Let b_u be the starting position, in sequence H, or read u. After transitive reduction, the event that two reads u, v from the same quasispecies are connected with an edge (u, v) is the event that (a) these two reads exist, and (b) no read w from the same quasispecies satisfies $b_u < b_w < b_v$.

Let us fix a quasispecies A. Given N reads, L positions in H and q quasispecies, the probability that a position is b_u for some read u of A is N/Lq. Assuming that b_u is such a position, there is a unique true edge (u, v) indicating an overlap with another read of A. The event that $b_v - b_u > k$ is the event that $b_u + 1, b_u + 2, \ldots, b_u + k$ are not beginnings of reads of A, and since the reads have uniformly random positions, the probability of that event is

$$p_k = \left(1 - \frac{N}{Lq}\right)^k \approx \exp(-kN/Lq)$$

The probability that $b_v - b_u = k$ is $p'_k = \frac{N}{Lq} p_{k-1}$.

If $b_v - b_u$ is much larger than Lq/N then most probably b_v is a read from another quasispecies B, and the reason for the difference is not a gap between the positions of various reads of A, but the fact that in the interval between b_u and b_v the sequences of quasispecies A and B are different. Therefore if $\Delta = b_v - b_u$, the number $1/p_\Delta \approx= \exp(\Delta N/Lq)$ measures the "implausibility" that (u, v) is a true edge. By "implausibility" we mean a quantity that is low when the edge is plausible and high when it is not.

If the lengths of the reads are variable and random, then after the cleaning we should have larger gaps following beginnings of particularly long reads. However, the distributions of lengths of the survivors of the cleaning process is more uniform (it has a much smaller variance) than the original distribution of read lengths. Thus we have a reasonable approximation.

If we have different frequencies of reads from different quasispecies, then the proper formula for quasispecies A would be $\exp(\Delta N f_A/L)$ rather than $\exp(\Delta N/Lq)$. However, We cannot use it because a-priori we do not know the frequencies.

What is least clear from our analysis is what function of p_Δ would give the best result if we use it as the cost of edge (b_u, b_v). We will try to natural candidates: the inverse, giving formula $\exp(\Delta N/Lq)$ and minus logarithm, giving formula $\Delta N/Lq$.

4 Quasispecies Assembly Via Network Flows

In this section we show how to modify the read graph G into a flow network so that Quasispecies Assembly would naturally represented by a network flow through G. We then reformulate the Quasispecies Assembly problems into the minimum-cost network flow problems.

As we noticed in Section 3.1, each quasispecies corresponds to a simple path in the (transitively reduced) read graph G. Each such path can be viewed as a *flow* originated in the source corresponding to the first read flowing through intermediate reads and ending at the sink corresponding to the last read.

Standard network flow formulations associate flow with the edges rather than the vertices. Therefore, our first modification to the read graph would be replacement of each vertex corresponding to a read r with the beginning vertex r_b and the ending vertex r_e connected with the edge (r_b, r_e). Each edges with the head v changes its head to r_b and each edge with the tail v changes it tail to r_e (see Figure 3).

For simplification of the flow formulation we also introduce universal source and sink vertices s and t for all flows (see Figure 4). We add an edge from the source s to each read that does not have any incoming edges and an edge to the sink t from each read that does not have outgoing edges. These two vertices are also supplied with back edge (t, s) through which each quasispecies flow should return back thus making our flow circular.

We now ready to formulate the minimum cost feasible flow problem corresponding to Most Parsimonious Quasispecies Assembly. Let $f : E \to R_0^+$ be a *circular flow* defined on all edges. The value of the flow f through a read edge (b_r, e_r) represents the number of quasispecies that contain the read r. The corresponding linear program (1-4) is as follows:

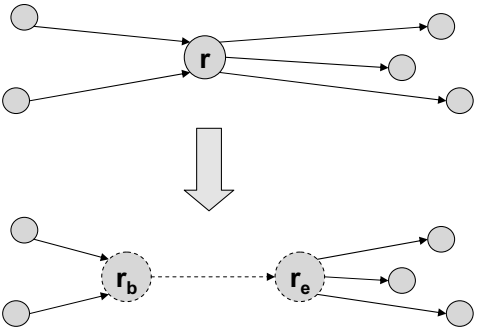


Fig. 3. Replacing of a vertex corresponding to a read r with the edge (r_b, r_e). The new vertices and edges are dashed.

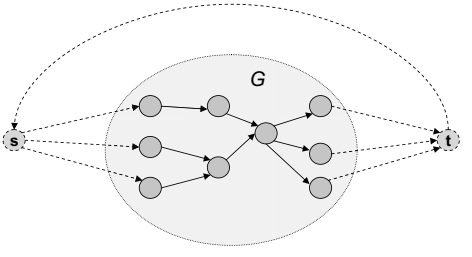


Fig. 4. Universal source s and universal sink t with the backward edge (t, s) are added to the read graph. The new vertices and edges are dashed.

$$\text{Minimize} \quad f(t,s) \tag{1}$$

subject to

$$\forall v \in V \quad \sum_{(u,v)\in E} f(u,v) = \sum_{(v,u)\in E} f(v,u) \tag{2}$$

$$\forall \text{read } r \in R \;\; f(b_r, e_r) \geq 1 \tag{3}$$

$$\forall (u,v) \in E \;\; f(u,v) \geq 0 \tag{4}$$

Objective (1) is parsimonious – it asks for minimizing number of quasispecies since each unit of flow corresponding to a single quasispecies should pass through the edge (t, s) exactly once. Constraint (2) is the flow conservation – for each vertex $v \in V$, the total flow entering v equals the total flow exiting v. Constraint (3) requires that each read to be covered by at least one predicted quasispecies. Constraint (4) forbids the backward flow so that the flow would really correspond to quasispecies.

The linear program (1-4) does not predict complete quasispecies but rather decides which pairs of overlapping reads belong to the same quasispecies. In order to obtain a feasible set of quasispecies one can simply replace each edge e with $f(e)$ copies and in the resulted graph find $f(t, s)$ edge-disjoint $s - t$-paths each corresponding to a quasispecies.

Although the linear program (1-4) does not require flow to be integer, the optimal integer solution is always fractionally optimal. All linear program solvers (e.g., [13,12]) find optimal integer solutions very efficiently. Alternatively, one can use a faster combinatorial min-cost flow solver [11,14].

The next linear program solves Minimum Cost Quasispecies Assembly. Here, we set to zero the cost of all edges introduced into G while modifying it into the flow network, i.e., $cost(t,s) = 0$, $cost(s,u) = cost(v,t) = 0$ and, for each read r, $cost(r_b, r_e) = 0$.[1] The only difference with the previous formulation is in the objective. Objective (5) does not pay attention to the number of predicted quasispecies but to the total cost of all predicted quasispecies.

$$\begin{aligned} &\text{Minimize} \quad \sum_{e \in E} cost(e) \cdot f(e) && (5)\\ &\text{subject to} \\ &\forall v \in V \quad \sum_{(u,v) \in E} f(u,v) = \sum_{(v,u) \in E} f(v,u) \\ &\forall \text{read } r \in R \;\; f(b_r, e_r) \geq 1 \\ &\forall (u,v) \in E \;\; f(u,v) \geq 0 \end{aligned}$$

Finally, Minimum Cost Parsimonious Quasispecies Assembly is solved with the same linear program as Minimum Cost Quasispecies Assembly. The only difference is that $cost(t,s)$ is set to a very large number. As a result any feasible assembly cannot be optimal if it uses more than the minimum possible number of quasispecies and as a secondary criteria the total cost of read overlaps is minimized.

5 Experimental Results

To our knowledge, full-genome quasispecies data for HCV is currently unavailable. However, previous research has obtained the sequences of individual HCV quasispecies for several important subregions. [1] obtained quasispecies data for the E1E2 region of the HCV genome. This data is a contiguous region 1734 bp long over $Q = 44$ quasispecies. We generated two simulated problem instances based upon the sequences in this data. The first instance, which we shall refer to as the "uniform" instance, assumed that the frequencies of each of the sequences in the data set were equal all equal (i.e. $f(q_i) = 1/Q$ for all q_i). The second instance, which we refer to as the "nonuniform" instance, assumed that the frequency of one quasispecies was $1/2$, while all other quasispecies had equal frequency of $1/(2(Q-1))$.

We assumed that the 454 Lifesciences system would produce approximately 250K reads of length 100 across the entire 9.6K bp length of the HCV genome. Since the E1E2 region is 1.7K bp long, approximately 18% of the 250K reads (approx. 44K) reads should span the E1E2 region. Reads were generated by iteratively selecting a sequence

[1] Alternatively, the cost of the read can be set inversely proportional to the number of copies of the read. This way the multiplicity of the read participation in assembly should correspond to its multiplicity among collection of all reads.

Instance	# of reads	# of reads after cleaning	# of overlaps	# of overlaps after transitive reduction
Uniform	44028	7810	1047340	19664
Nonuniform	44029	6097	674082	16350

Fig. 5. This table shows the original number of reads, the number of reads after "cleaning" (i.e. removing subreads), and the number of overlaps (i.e. number of edges in the read graph) for the two problem instances considered. The "uniform" problem instance consisted of 44 quasispecies of length 1734 each with equal frequency. The "nonuniform" instance consisted of the same 44 quasispecies, but one quasispecies was selected to have frequency $1/2$ and all others were given frequency $1/(2(Q-1))$.

from E1E2 at random according to that read's frequency and fragmenting it into reads which were then accumulated in a collection; the lengths of reads were generated using a normal distribution with $\mu = 100, \sigma^2 = 10$.

Once a problem instance was generated by the above procedure, we removed reads that were contained within other reads. The reason we introduce this "cleaning" phase of our algorithm is two-fold: first, any read that is subread of another cannot possibly introduce a new quasispecies, and second, the graph formed by the remaining reads is guaranteed to be acyclic, connected, and has a single global source and sink. Due to the large degree of homogeneity between quasispecies, a surprisingly large number of reads are cleaned out of the problem instance. After cleaning the problem instance of subreads, the read graph is constructed. The table on Figure 5 gives the various parameters for each of the two problem instances under consideration.

Out of the many possible overlaps between reads in the problem instance, only a small portion actually belongs to real quasispecies. From the table on Figure 6 one can see how well our min-cost flow based algorithms for Most Parsimonious Quasispecies Assembly and Minimum Cost Quasispecies Assembly predict which overlaps are true overlaps. The most parsimonious solution obtained by setting to 1 the back edge cost while other edges has cost 0 – in the table the corresponding solution is placed the row with cost function 1. We run the min-cost flow algorithm for the two problem instances under the following two different edge-cost functions. The cost function Δ equals the the difference in genome offsets of edge tail and head reads. This function is proportional to the logarithm of the estimated probability of the read overlap to be true overlap. As a result, the total cost of a path is proportional to the probability of it to be a true quasispecies. The cost function e^{Δ} is the estimated probability of the corresponding overlap to be true overlap.

The table on Figure 6 gives the total number of true overlaps and the total number of predicted overlaps. Then we give the number of true and false overlaps among predicted overlaps as well as the number of true overlaps which are missed by our method. Our experiments show that Most Parsimonious Assembly is the furthest from the true quasispecies and that the exponential cost is superior to the Δ-cost.

Similarly to the error measure for diploid organism phasing, we introduce the *switching error*, which is computed as follows. For each true quasispecies, we identify the path in the transitively reduced read graph and count how many times that path switches

Instance	Cost Function	True overlaps	Predicted overlaps	Correctly predicted overlaps	True, unpredicted overlaps	Incorrectly predicted overlaps	Switching Error	
							Lower bound	Random walk
Uniform	e^{Δ}	8048	8038	8005	43	33	1.07	44.5
	Δ		8145	7742	306	403	13.2	54.5
Nonuniform	e^{Δ}	6338	6328	6288	50	40	1.78	41.9
	Δ		6387	5974	364	413	15.5	50.1

Fig. 6. The number of real, predicted, correctly predicted, incorrectly predicted, and unpredicted overlaps for the two instances and three network flow methods (cost functions). Minimum Cost Quasispecies Assembly are denoted by cost Δ and e^{Δ}.

Instance	Runtime (in seconds)		
	Cleaning	Transitive reduction	Linear Program
Uniform	29.18	71.36	80.91
		75.53	58.24
Nonuniform	14.9	35.8	51.56
		36.65	35.61

Fig. 7. The runtimes for major subroutines in the program. All runtimes were recorded on a modern machine with an Intel Core Duo 2 CPU and 2 GB of RAM.

between predicted quasispecies and then average number of switches over all true quasispecies. The lower bound is the average number of unpredicted overlaps that occur along the path corresponding to a real quasispecies. We split the flow into quasispecies by walking from the universal source to the universal sink, randomly choosing which edge of each fork to tranverse, decrementing the flow along each edge as it is traversed. The column *Random Walk* in 6 reports the average switching distance over all true quasispecies for the set of randomly predicted quasispecies. Our results show that the exponential cost is superior to Δ-cost as well as the most parsimonious solution and that our method admits only very small fraction of possible errors.

Obviously, randomly predicting quasispecies has an high switching error. By using a more intelligent path-splitting heuristic, one can possibly reduce the switching error down to the lower bound.

The table on Figure 7 gives the runtimes for the instance cleaning, transitive reduction, and linear programming subroutines in the program. As the table indicates, our method can deliver results in a reasonable amount of time, and is expected to scale well to the sizes of real problem instances.

References

1. Von Hahn, T., Yoon, J.C., Alter, H., Rice, C.M., Rehermann, B., Balfe, P., Mckeating, J.A.: Hepatitis C Virus Continuously Escapes From Neutralizing Antibody and T-Cell Responses During Chronic Infection In Vivo. Gastroenterology 132, 667–678 (2007)
2. Myers, G.: Building Fragment Assembly String Graphs. In: European Conf. on Computational Biology, pp. 79–85 (2005)

3. Lippert, R., Schwartz, R., Lancia, G., Istrail, S.: Algorithmic strategies for the single nucleotide polymorphism haplotype assembly problem. Briefings in Bioinformatics 3(1), 23–31 (2002)
4. Alekseyev, M.A., Pevzner, P.A.: Colored de Bruijn graphs and the genome halving problem. IEEE/ACM Trans Comput Biol Bioinform. 4(1), 98–107
5. Chaisson, M.J., Pevzner, P.A.: Short read fragment assembly of bacterial genomes. Genome research (to appear, 2007)
6. Sundquist, A., Ronaghi, M., Tang, H., Pevzner, P., Batzoglou, S.: Whole-genome sequencing and assembly with high-throughput, short-read technologies. PLoS ONE 2(5), e484 (2007)
7. Brinza, D., Zelikovsky, A.: 2SNP: Scalable Phasing Based on 2-SNP Haplotypes. Bioinformatics 22(3), 371–373 (2006)
8. 454 Lifescience (2007), http://www.454.com/
9. Margulies, M., et al.: Genome sequencing in microfabricated high-density picolitre reactors. Nature 437(7057), 376–380 (2005)
10. Albert, R., DasGupta, B., Dondi, R., Sontag, E., Zelikovsky, A., Westbrooks, K.: Signal Transduction Network Inference from Indirect Experimental Evidence. Journal of Computational Biology 14(7), 927–949 (2007)
11. Goldberg, A.: An Effcient Implementation of a Scaling Minimum-Cost Flow Algorithm. Journal of Algorithms 22(1), 1–29 (1997)
12. GNU Linear Programming Kit, http://www.gnu.org/software/glpk/
13. ILOG CPLEX, http://www.ilog.com/products/cplex/
14. IG Systems CS2 Software (2007), http://www.igsystems.com/cs2/
15. Venter, J.C., Remington, K., Heidelberg, J.F., Halpern, A.L., Rusch, D., Eisen, J.A., Wu, D., Paulsen, I., Nelson, K.E., Nelson, W., et al.: Environmental genome shotgun sequencing of the Sargasso Sea. Science 304, 66–74 (2004)

A Dynamic Programming Algorithm for De Novo Peptide Sequencing with Variable Scoring

Matthew A. Goto[1] and Eric J. Schwabe[2]

[1] School of CTI, DePaul University, Chicago, IL 60604
mattgoto@gmail.com
[2] School of CTI, DePaul University, Chicago, IL 60604
eschwabe@cs.depaul.edu

Abstract. In the de novo peptide sequencing problem, output data from a tandem mass spectrometer are used to determine the peptide whose fragmentation yielded the output. Candidate peptides can be determined by finding forbidden-pairs paths in a spectrum graph constructed from the mass spectrometer data, assigning scores to vertices and/or edges in order to favor higher-scoring peptides. Chen et al. gave an algorithm to find the highest-scoring forbidden-pairs path in such a graph. However, in some scoring models, a vertex's score may vary depending on which incident edges are used in the path containing the vertex, ruling out the use of this algorithm. We give an algorithm to solve the highest-scoring forbidden-pairs paths problem when vertex scores can vary depending on the incident edges used that runs in $O(n^2d^3)$ time on a graph with n forbidden pairs and a maximum vertex degree of d, and prove its correctness. We are currently working on a Java implementation of this algorithm that we plan on incorporating into the Illinois Bio-Grid Desktop.

1 Introduction

In the field of proteomics, the problem of identifying the structure of an unknown peptide in a sample is central. In particular, one may be given tandem mass spectrometer (MS/MS) output and want to determine the structure of the peptide that yielded that output. There are two commonly used approaches to finding the original peptide: *database matching*, where the best match is found from a database of outputs generated by known peptides; and *de novo sequencing*, where the peptide is determined solely from the MS/MS output.

While the databases of known peptides are growing, the database matching approach still has some drawbacks. First, it requires additional information (e.g., the genome from which the peptide came) in order to be able to reliably determine the input peptide, and this additional information is not always available. Second, on a peptide that has never been identified before, a database search is likely to be fruitless. Thus de novo sequencing remains important even when databases are available. (See, e.g., Bafna and Reinert [1], Lu and Chen [7].)

I. Măndoiu, R. Sunderraman, and A. Zelikovsky (Eds.): ISBRA 2008, LNBI 4983, pp. 171–182, 2008.

An output spectrum generated by a tandem mass spectrometer consists of a list of peaks. Each peak has an *intensity* representing its abundance in the output, and a *value* representing its mass/charge ratio. Such a spectrum is typically the end result of the application of a series of chemical and mechanical techniques:

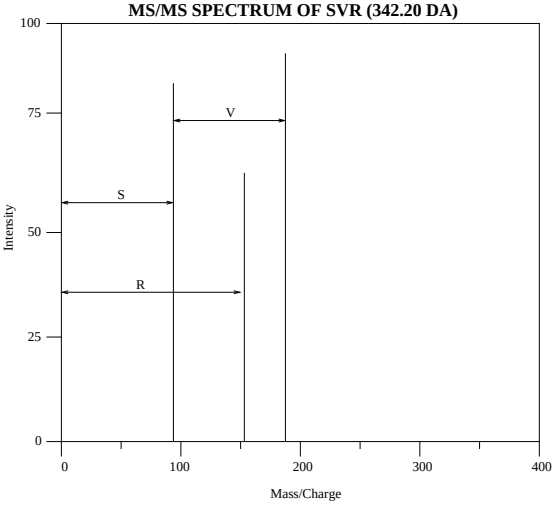


Fig. 1. A hypothetical MS/MS spectrum for the peptide SVR

First, a protein is digested to yield a collection of peptides. As there may be many different peptides in the digested sample, some technique for separating the peptides (such as liquid chromatography) is applied.

The resulting sample is then put through a mass spectrometer, which determines the mass/charge ratio and abundance of each peptide in the sample. A sufficiently abundant peptide is selected and fragmented further by another process (such as collisionally-induced dissociation) that often breaks the peptide molecules between adjacent amino acids. The charge and composition of the peptide determines where these breakages occur.

There are many types of fragment ions that can occur as a result of the fragmentation process. When the fragmentation occurs between two amino acids in the original peptide, prefix and suffix ions of the peptide (called, respectively, b-ions and y-ions) are created. These are the most common types of fragments, but there are others, including the products of neutral losses and fragments that result from breakages that are not at the boundaries between amino acids.

The sample consisting of the various ions resulting from the fragmentation of the peptide is put through a second mass spectrometer. The output of the second mass spectrometer is a *tandem mass spectrometer (MS/MS) spectrum*. The MS/MS spectrum, consisting of a set of (mass/charge, intensity) pairs for the

fragments of the peptide, is used as the input to the de novo peptide sequencing problem. In addition to the list of pairs, the total mass and charge of the peptide (as determined by the first mass spectrometer run) are also given.

The *de novo sequencing problem* is to find the peptide that served as the input to the first mass spectrometer run, given only the MS/MS spectrum and the mass and charge of the peptide. The problem is complicated by the fact that MS/MS spectra are often noisy – that is, they may contain peaks that are not generated by actual fragments of the peptide but rather have some other cause, such as a contaminant or machine error – and not all possible fragmentations between adjacent amino acids may have occurred.

Section 2 will discuss previous work on the de novo sequencing problem and how the problem of finding the highest-scoring forbidden-pairs path with variable vertex scores arose from this work. Section 3 will present our dynamic programming algorithm and give a sketch of a proof of its correctness. Section 4 will briefly discuss our implementation of the algorithm and future work.

2 Previous Work

A common approach to the de novo peptide sequencing problem is to construct a graph from a subset of the peaks in the spectrum, and find some path in the graph that corresponds to a peptide. The goal is to find the highest-scoring path, given some set of vertex and/or edge scores that are chosen so that higher-scoring paths correspond to more likely peptides.

2.1 Spectrum Graphs

A *spectrum graph* (see, e.g., Dancik et al. [3]) is constructed by creating a vertex for each peak in the MS/MS spectrum and for each type in a chosen set of k fragmentation ion types (e.g., b-ion, y-ion, etc.). For each $i \in 1, \ldots, k$, let δ_i be the fixed amount by which the mass of an ion of the ith fragmentation type would differ from the sum of the masses of the amino acids in that fragment. Then for each peak in the spectrum with mass p, k vertices are created having masses of $p + \delta_i$, one for each fragmentation ion type $i \in 1, \ldots, k$.

There are two additional vertices in the spectrum graph: one with a mass of 0 and one with the original mass of the peptide. If two or more vertices are created with the same mass, they can be merged into a single vertex.

A directed edge connects two vertices in the spectrum graph if and only if the difference between the vertices' masses is the mass of a single amino acid. (Some constructions include edges between vertices whose masses differ by the sum of two or more amino acids.) All edges are directed from a smaller mass to a larger mass, yielding a directed acyclic graph.

For an example, see Figures 1 and 2, where the construction uses the two fragmentation ion types of b-ion and y-ion so that the ith peak in the spectrum generates two vertices x_i and y_i, and two pairs of vertices have been merged.

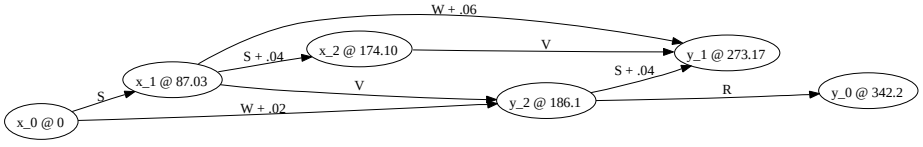


Fig. 2. A spectrum graph constructed from Figure 1

2.2 Forbidden-Pairs Paths

Although each peak in the spectrum graph may produce several vertices, at most one of these vertices can be included in any path through the spectrum graph that represents a peptide, since the vertices represent mutually exclusive interpretations of the peak. These sets of vertices are called *forbidden sets*, or in the case where only two fragmentation ion types are used, *forbidden pairs*. A path that uses at most one vertex from each forbidden pair is called a *forbidden-pairs path* (Dancik et al. [3] called this an *antisymmetric path*). Each such path represents a peptide that may have generated the MS/MS spectrum.

Chen et al. [2] gave a dynamic programming algorithm to solve the highest-scoring forbidden-pairs path problem for a spectrum graph. When constructing the spectrum graph, they used the two fragmentation ion types of b-ion and y-ion, yielding forbidden pairs of vertices. Their algorithm is as follows, where E is the adjacency matrix of the graph and s is the edge scoring function[1]:

Algorithm Compute-$Q(G)$
1. Initialize $Q(i,j) = 0$ for all $0 \leq i, j \leq n$;
2. For $j = 1$ to n
3. If $E(y_j, y_0) = 1$, then $Q(0,j) = \max\{Q(0,j), s(y_j, y_0)\}$;
4. If $E(x_0, x_j) = 1$, then $Q(j,0) = \max\{Q(j,0), s(x_0, x_j)\}$;
5. For $i = 0$ to $j - 1$

(a) For every $E(y_j, y_p) = 1$ and $Q(i,p) > 0$, $Q(i,j) = \max\{Q(i,j), Q(i,p) + s(y_j, y_p)\}$;

(b) For every $E(x_p, x_j) = 1$ and $Q(p,i) > 0$, $Q(j,i) = \max\{Q(j,i), Q(p,i) + s(x_p, x_j)\}$;

Fig. 3. Line 3: Starting a new pair of paths x_0 and y_j to y_0

The algorithm runs in $O(VE)$ time, where $V = 2n + 2$ is the number of vertices in the graph (n being the number of forbidden pairs in the graph), and E is the number of edges in the graph.

[1] In Line 5 of the algorithm, Chen et al. start at $i = 1$, but we believe this to be a typographical error.

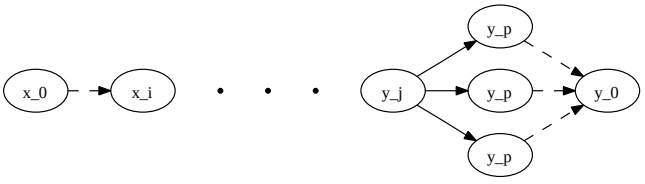


Fig. 4. Line 5a: Extending an existing pair of paths from $x_0 \ldots x_i$ and $y_p \ldots y_0$ to $x_0 \ldots x_i$ and $y_j \ldots y_0$

2.3 Fixed Scoring and Variable Scoring

Vertex and/or edge scores can be chosen to increase the likelihood of finding the correct peptide by influencing which forbidden-pairs path will be returned by an algorithm. In most scoring models, each vertex and/or edge is given a fixed score, and the scores along a path are added to determine the score of the path. Dancik et al. [3] gave a scoring function based on observations of the frequencies of various fragmentation ion types in spectra generated by known peptides that assigned "a premium for present ions [vertices] and a penalty for missing ions [vertices]." The algorithm of Chen et al. [2] works for an arbitrary scoring function that is supplied as input. (We note that the technique of scoring candidate peptides is not limited to the spectrum graph approach. Peptides are also assigned scores in database matching algorithms and in de novo algorithms that do not use spectrum graphs; see, e.g., Fisher et al. [4] and Mo et al. [8].)

In their paper on the PepNovo system [5], Pevzner and Frank proposed a scoring model based on conditional probabilities of different types of peptide fragmentation occurring in observed spectra of known peptides. Their model yields vertex scores that depend on which of the vertices' incident edges are used in the path. For this *variable vertex scoring* method, neither the algorithm of Dancik et al. nor that of Chen et al. can be applied to the problem. Pevzner and Frank stated that they used a variant of the Chen et al. algorithm in PepNovo, but the details were not discussed in the paper nor in the PepNovo source code.

3 Solving the Problem with Variable Vertex Scores

In this section, we present a new dynamic programming algorithm that finds the highest-scoring forbidden-pairs path in a spectrum graph with variable vertex scores, n forbidden pairs, and maximum vertex degree d in $O(n^2d^3)$ time, and prove its correctness. This algorithm is the first that has been proved to always find the highest-scoring forbidden-pairs path when vertices' scores depend on the incident edges chosen for each vertex in the path.

3.1 Transforming the Spectrum Graph

To accommodate variable vertex scores, we replace each vertex of the spectrum graph with a *super-vertex* containing a complete bipartite graph. This replacement transforms a spectrum graph in which the vertex scores depend on the incident edges used by a path into one with fixed edge scores (and no vertex scores). We can construct a highest-scoring forbidden-pairs path in the original graph from a highest-scoring forbidden-pairs path in the transformed graph (where the forbidden pairs are now actually forbidden pairs of super-vertices) by just taking the sequence of super-vertices visited.

Each bipartite graph consists of a *left set* of vertices with a vertex for each incoming edge of the original vertex, and a *right set* of vertices with a vertex for each outgoing edge of the original vertex. In the super-vertex for v, we call these sets $L(v)$ and $R(v)$. $L(x_0)$ and $R(y_0)$ are each defined to consist of a single vertex, named s and t, respectively. All edges in the complete bipartite graph are directed from left to right.

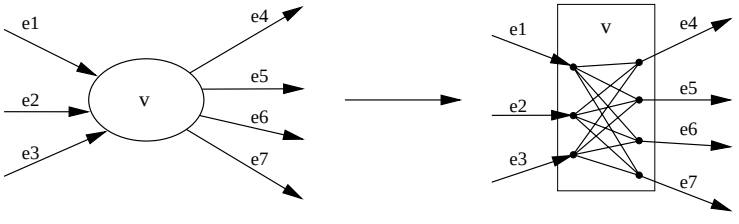


Fig. 5. The transformation of a vertex with a variable score to a super-vertex containing a bipartite graph

To find the score for the edge from u' in $L(B)$ to v' in $R(B)$ within a super-vertex B: Let A be the super-vertex such that u' is the lone successor of some vertex in $R(A)$. Let C be the super-vertex such that v' is the lone predecessor of some vertex in $L(C)$. Then the score of (u', v'), called $w(u', v')$, is defined to be the score of the original vertex B when the incident edges used are (A, B) and (B, C). For each edge (U, V) in the original graph with score x, the edge from a vertex $u' \in R(U)$ to a vertex $v' \in L(V)$ will be given score $w(u', v') = x$ also.

3.2 Our Dynamic Programming Algorithm

Previous algorithms for finding forbidden-pairs paths cannot be applied to solve the problem on graphs with variable vertex weights. They also cannot be applied to a transformed spectrum graph, as there are no longer the disjoint forbidden pairs that these algorithms require – only forbidden pairs of super-vertices.

The following algorithm finds the highest-scoring forbidden-pairs path in the transformed spectrum graph using dynamic programming, extending the approach of Chen et al. [2], as follows. We complete a two-dimensional table Q

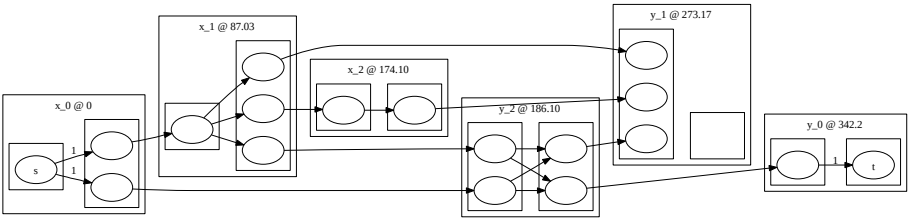


Fig. 6. The result of transforming the spectrum graph in Figure 2

of objects $Q(i,j), 0 \leq i,j \leq n$ where n is the number of forbidden pairs in the original spectrum graph. Each $Q(i,j)$ contains:

- maxWeight: the maximum score of all possible pairs of paths from s in $L(x_0)$ to some u in $L(x_i)$ and from some v in $R(y_j)$ to t in $R(y_0)$ that contain at most one super-vertex from each forbidden pair.
- weights: a map with key set a subset of $L(x_i) \times R(y_j)$, where weights(u,v) is an object containing three things:
  - score: the score of the max-score paths from s to u and from v to t that contain at most one super-vertex from each forbidden pair.
  - backtrackIndices: a pair of indices in Q which will be one of $(0,0), (i,p)$, or (p,j). This is used for backtracking.
  - backtrackKey: a key into the weights map from $Q(0,0), Q(i,p)$, or $Q(p,j)$ depending upon if the pair of indices is $(0,0)$, (i,p), or (p,j). This is also used for backtracking.

  If no paths from s to u and from v to t that contain at most one super-vertex from each forbidden pair have been discovered, or none exist, weights(u,v) is not assigned a value in the map.

The table is constructed as follows:

1. For all i and j between 0 and n
   $Q(i,j)$.maxWeight $= -\infty$ (except for $Q(0,0)$.maxWeight $= 0$)
   $Q(i,j)$.weights $= \{\}$ (but $Q(0,0)$.weights $= \{((s,t),(0,(0,0),(s,t)))\}$)
2. For $j = 1$ to n
3. If an edge exists between some vertex u in $R(y_j)$ and v in $L(y_0)$ and $w(u,v) + w(v,t) > Q(0,j)$.maxWeight, set
   $Q(0,j)$.maxWeight $= w(u,v) + w(v,t)$
   $Q(0,j)$.weights$(s,u) = (w(u,v) + w(v,t),(0,0),(s,t))$
   (See Figure 7.)
4. If an edge exists between some vertex u in $R(x_0)$ and v in $L(x_j)$ and $w(s,u) + w(u,v) > Q(j,0)$.maxWeight, set
   $Q(j,0)$.maxWeight $= w(s,u) + w(u,v)$
   $Q(j,0)$.weights$(v,t) = (w(s,u) + w(u,v),(0,0),(s,t))$
5. For $i = 0$ to $j - 1$

a. For each u in $L(x_i)$ and v in $R(y_j)$, compute $Q(i,j)$.weights(u,v) as follows:

For every super-vertex y_p such that there is an edge from v to some a in $L(y_p)$ and such that $Q(i,p)$.maxWeight $> -\infty$, compute the max over all b in $R(y_p)$ such that $Q(i,p)$.weights(u,b) exists of $Q(i,p)$.weights$(u,b) + w(v,a) + w(a,b)$. If there are no such y_p or b, (u,v) is not given a value in the map at this time. Otherwise, call the resulting maximum uvWeight, and let (u,b^*) be the key to $Q(i,p)$.weights that produces this maximum value. $Q(i,j)$.weights$(u,v) =$ (uvWeight,$(i,p),(u,b^*)$). (See Figure 8.)

If any keys in $Q(i,j)$.weights have a value, then do the following: Let (u^*,v^*) be the key of the largest value of score in the map $Q(i,j)$.weights. Set $Q(i,j)$.maxWeight $= Q(i,j)$.weights(u^*,v^*).score.

b. For each u in $L(x_j)$ and v in $R(y_i)$, compute $Q(j,i)$.weights(u,v) as follows:

For every super-vertex x_p such that there is an edge from some a in $R(x_p)$ to u and such that $Q(p,i)$.maxWeight $> -\infty$, compute the max over all b in $L(x_p)$ such that $Q(p,i)$.weights(b,v) exists of $Q(p,i)$.weights$(b,v) + w(b,a) + w(a,u)$. If there are no such x_p or b, (u,v) is not given a value in the map at this time. Otherwise, call the resulting maximum uvWeight, and let (b^*,v) be the key to $Q(p,i)$.weights that produces this maximum value. $Q(j,i)$.weights$(u,v) =$ (uvWeight,$(p,i),(b^*,v)$).

If any keys in $Q(j,i)$.weights have a value, then do the following: Let (u^*,v^*) be the key of the largest value of score in the map $Q(j,i)$.weights. Set $Q(j,i)$.maxWeight $= Q(j,i)$.weights(u^*,v^*).score.

To get the score of the highest-scoring path from Q, called $W(i,j)$, use the following algorithm:

6. For $i = 0$ to n
7. For $j = 0$ to n
8. If there is an edge (a,b) from vertex a in $R(x_i)$ to vertex b in $L(y_j)$ and $Q(i,j)$.weights has more than zero keys, compute
   $W(i,j)$.pathWeight = the maximum of
   $Q(i,j)$.weights(u,v).score $+ w(u,a) + w(a,b) + w(b,v)$ over all keys (u,v) in $L(x_i) \times R(y_j)$ for which $Q(i,j)$.weights(u,v) exists.
   $W(i,j)$.backtrack $= (u^*,v^*)$ where (u^*,v^*) is the key to $Q(i,j)$.weights that maximized the value of $W(i,j)$.pathWeight.
   Otherwise (if there is no (a,b) edge or there are zero keys in $Q(i,j)$.weights),
   $W(i,j)$.pathWeight and $W(i,j)$.backtrack are given no value.
   (See Figure 9.)
9. The largest value of $W(i,j)$.pathWeight, found at location (i^*,j^*), is the score of the highest-scoring forbidden-pairs path from s to t. If no $W(i,j)$.pathWeight has a value, there are no forbidden-pairs paths.

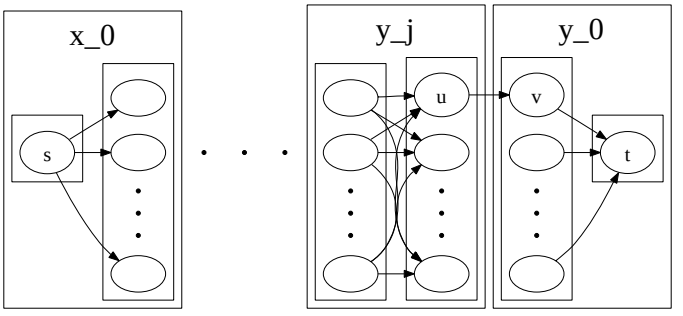


Fig. 7. Line 3: Starting a new pair of paths x_0 and y_j to y_0. Similar to Figure 3.

Backtracking to compute the highest-scoring forbidden-pairs path is relatively straightforward, going from (i^*, j^*) to $(0, 0)$ using $W(i^*, j^*)$.backtrack and the backtrackIndices and backtrackKey fields of the $Q(i, j)$.weights references. (Details will appear in the full paper.)

3.3 Running Time Analysis and Proof of Correctness

Let d be the maximum vertex degree of the original spectrum graph. Step 1 takes $\Theta(n^2)$ steps, as there are n^2 elements in the table Q. Steps 3 and 4 are each executed n times, and each execution takes $O(d)$ steps, assuming that the transformed graph is represented as an adjacency matrix, and that get and put operations for the weights map take constant time. Steps 5a and 5b are each executed $\Theta(n^2)$ times, and each execution takes $O(d^3)$ steps, as for each u and v there are up to d values of b over which we must take the maximum path score. Finally, Steps 6-9 take a total of $O(n^2d^2)$ steps, as there are $\Theta(n^2)$ iterations each taking $O(d^2)$ steps. The dominant term from Steps 5a and 5b yields an overall running time of $O(n^2d^3)$.

The correctness of the algorithm follows directly from the following theorem.

Theorem: For any i and j between 0 and n, where $i < j$, and for $(i, j) = (0, 0)$, the following eight-part predicate $M(i, j)$ holds:

1. When the object $Q(i, j)$ is computed by the algorithm above, we have that
   (a) for every u in $L(x_i)$ and v in $R(y_j)$, weights(u, v).score is the value of the maximum possible score of paths from s to u and from v to t that contain at most one super-vertex from each forbidden pair (if any such paths exist – otherwise it has no value), and
   (b) for every u in $L(x_i)$ and v in $R(y_j)$, weights(u, v).backtrackIndices is the pair of indices in Q which is one of $(0, 0)$, (i, p), or (p, j) and results in the maximum possible value for weights(u, v).score (if weights(u, v).score is given a value – otherwise it has no value), and

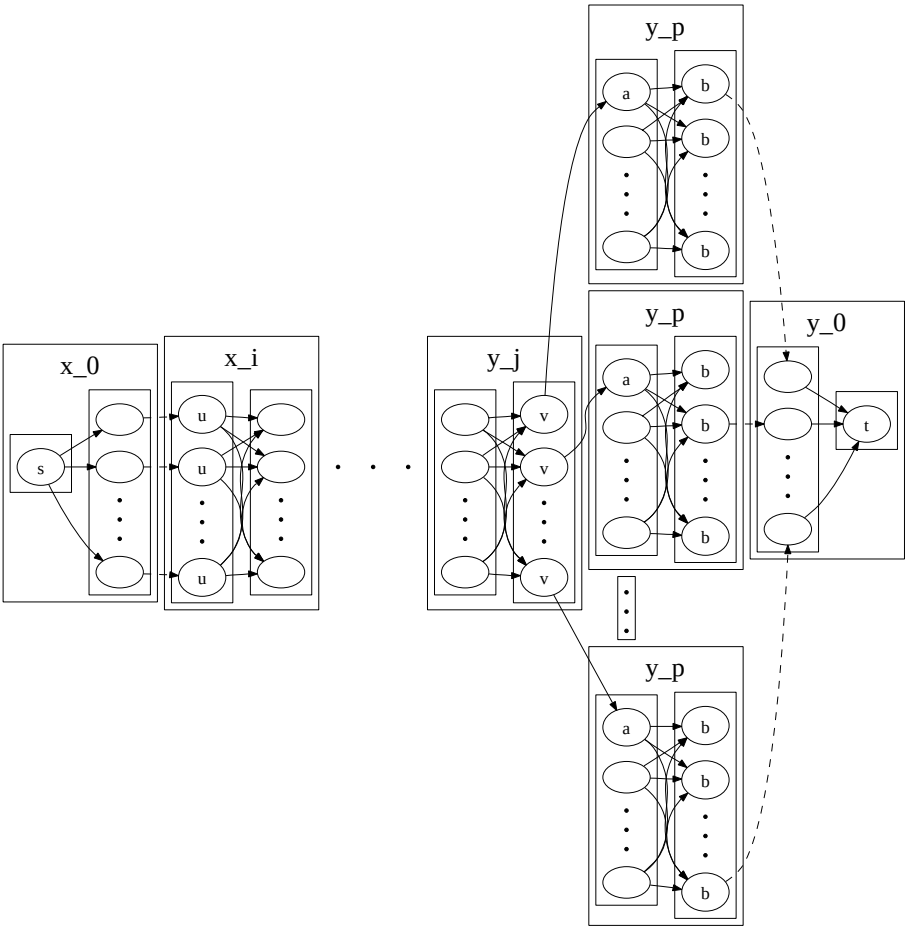


Fig. 8. Line 5a: Extending an existing pair of paths from $x_0 \dots x_i$ and $y_p \dots y_0$ to $x_0 \dots x_i$ and $y_j \dots y_0$. Similar to Figure 4.

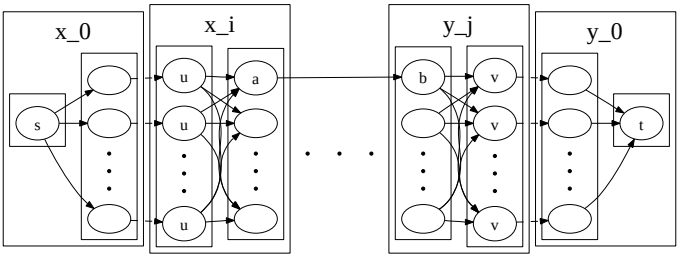


Fig. 9. Line 8a: Crossing over from x_i to y_j

(c) for every u in $L(x_i)$ and v in $R(y_j)$, weights(u, v).backtrackKey is the key into the weights map of Q(weights(u, v).backtrackIndices) which results in the maximum possible value for $Q(i, j)$.weights(u, v).score (if weights(u, v).score is given a value – otherwise it has no value), and
(d) maxWeight is the value of the maximum possible score of paths from s to any vertex in $L(x_i)$ and from any vertex in $R(y_j)$ to t that contain at most one super-vertex from each forbidden pair (if any such paths exist – otherwise it will be 0).

2. When the object $Q(j, i)$ is computed by the algorithm above, we have that
(a) for every u in $L(x_j)$ and v in $R(y_i)$, weights(u, v).score is the value of the maximum possible score of paths from s to u and from v to t that contain at most one super-vertex from each forbidden pair (if any such paths exist – otherwise it has no value), and
(b) for every u in $L(x_j)$ and v in $R(y_i)$, weights(u, v).backtrackIndices is the pair of indices in Q which is one of $(0, 0)$, (i, p), or (p, j) and results in the maximum possible value for weights(u, v).score (if weights(u, v).score is given a value – otherwise it has no value), and
(c) for every u in $L(x_j)$ and v in $R(y_i)$, weights(u, v).backtrackKey is the key into the weights map of Q(weights(u, v).backtrackIndices) which results in the maximum possible value for $Q(i, j)$.weights(u, v).score (if weights(u, v).score is given a value – otherwise it has no value), and
(d) maxWeight is the value of the maximum possible score of paths from s to any vertex in $L(x_j)$ and from any vertex in $R(y_i)$ to t that contain at most one super-vertex from each forbidden pair (if any such paths exist – otherwise it will be 0).

(While Parts 1 and 2 are nearly identical, they differ in that one addresses the part of the table above the main diagonal, and the other the part below.)

Sketch of Proof: The proof proceeds by induction on (i, j); the complete proof will appear in the full paper.

Base Case: The base case follows immediately from the initialization of $Q(0, 0)$.

Inductive Step: Let i and j between 0 and $n, i < j$, be given. Assume that $M(i, j)$ holds for all (i', j') where either $j' < j$ or ($j' = j$ and $i' < i$). This assumption covers all objects $Q(i', j')$ and $Q(j', i')$ that are computed before $Q(i, j)$ and $Q(j, i)$. 1(a)-(d) and 2(a)-(d) follow directly from the inductive hypothesis and the descriptions of Steps 3-5 of our algorithm.

Together, the base case and inductive step establish that $M(i, j)$ holds for all i and j between 0 and n, where $i < j$, and for $(i, j) = (0, 0)$. This establishes the theorem.

It follows from the theorem and the description of Step 8 of our algorithm that the largest of the at most $(n+1)^2$ values of $W(i, j)$.pathWeight must be the highest possible score of a forbidden-pairs path. If $W(i^*, j^*)$.pathWeight is the largest pathWeight in W, backtracking will construct a highest-scoring forbidden-pairs path from x_0 to y_0 in the original graph that passes through x_{i^*} and y_{j^*}.

4 Algorithm Implementation and Future Work

We have implemented our algorithm as part of a Java implementation of the PepNovo system developed by Frank and Pevzner [5], which we hope to eventually incorporate into the Illinois Bio-Grid Desktop [9] suite of bioinformatics tools. Our implementation was reverse-engineered from Frank and Pevzner's description of their scoring algorithm and publically available C++ source code. We have established, by consulting the source code and its authors, that our algorithm is not the algorithm used in PepNovo. To the best of our knowledge, the algorithm used in PepNovo has not been published elsewhere nor had its correctness proved.

Although we have finished the implementation of our algorithm, our incorporation of it into a full implementation of the PepNovo system is still in progress. Since our contribution is not the development of a new scoring function, but rather the description of an algorithm that can be rigorously shown to always find the highest-scoring forbidden-pairs path when variable vertex scores are allowed, we do not expect our implementation to perform significantly differently than PepNovo unless there are cases where our algorithm finds a highest-scoring path that the algorithm implemented by Frank and Pevzner does not.

References

1. Bafna, V., Reinert, K.: Mass Spectrometry and Computational Proteomics. In: Encyclopedia of Genetics, Genomics, Proteomics and Bioinformatics, Wiley, Chichester (2005)
2. Chen, T., Kao, M.-Y., Tepel, M., Rush, J., Church, G.M.: A Dynamic Programming Approach to De Novo Peptide Sequencing via Tandem Mass Spectrometry. Journal of Computational Biology 8(3), 325–337 (2001)
3. Dancik, V., Adonna, T.A., Clauser, K.R., Vath, J.E., Pevzner, P.A.: De Novo Peptide Sequencing via Tandem Mass Spectrometry. Journal of Computational Biology 6(3/4), 327–342 (1999)
4. Fisher, B., Roth, V., Roos, F., Grossmann, J., Baginsky, S., Widmayer, P., Gruissem, W., Buhmann, J.M.: NovoHMM: A Hidden Markov Model for de Novo Peptide Sequencing. Analytic Chemistry 77, 7265–7273 (2005)
5. Frank, A., Pevzner, P.: PepNovo: De Novo Peptide Sequencing via Probabilistic Network Modeling. Analytical Chemistry 77, 964–973 (2005)
6. Kinter, M., Sherman, N.E.: Protein Sequencing and Identification Using Tandem Mass Spectrometry. Wiley-Interscience, Chichester (2000)
7. Lu, B., Chen, T.: Algorithms for de novo peptide sequencing using tandem mass spectrometry. Drug Discovery Today: BIOSILICO 2(2), 85–90 (2004)
8. Mo, L., Dutta, D., Wan, Y., Chen, T.: MSNovo: A Dynamic Programming Algorithm for de Novo Peptide Sequencing via Tandem Mass Spectrometry. Analytical Chemistry 79, 4870–4878 (2007)
9. Steele, A., Angulo, D.: The Illinois Bio-Grid: A Prototype for Industry-Academe Collaboration. In: Proceedings of the 2003 Midwest Software Engineering Conference (2003)

Invited Keynote Talk: Haplotype Sharing for Genome-Wide Case-Control Association Studies

Andrew S. Allen

Department of Biostatistics and Bioinformatics, DUMC 3850
Duke University, Durham, NC 27710
andrew.s.allen@duke.edu

The computational demands imposed by the large number of markers considered in a genome-wide association study (GWAS) have resulted in an extreme simplification in the types of analyses conducted. While sophisticated methodology may be used to adjust for multiple comparisons, most studies are analyzed one marker at a time using simple tests such as the Cochran-Armitage trend test. Though computationally simple, a significant limitation of such an analysis paradigm is its failure to efficiently use the information contained in the GWAS sample.

Haplotype sharing is a simple concept that attempts to translate between population genetics and genetic epidemiology. For recent mutations that cause disease, we would expect that haplotypes of case participants would be more similar to each other in the immediate region of the mutation than they would be to the haplotypes of control participants. This analysis is carried out without specifying the underlying evolutionary history that may have given rise to this sharing pattern, by using an ad-hoc definition of sharing between two haplotypes such as the number of loci up- and down-stream from a test locus that are identical by state.

In this talk, we outline the development of computationally simple association tests based on haplotype sharing that can be easily applied to case-control studies on the genome-wide scale. We give tests that allow for the use of fast (but not likelihood-based) haplotyping algorithms such as 2-SNP (http://alla.cs.gsu.edu/ software/2SNP/), while properly accounting for the statistical uncertainty introduced by using inferred or imputed haplotypes. Finally, we also consider the effect of covariates on haplotype sharing analyses, as many GWAS are analyzed using covariates to adjust for the potentially confounding effects of population stratification. The methods are illustrated by analyzing a whole genome association study of Parkinson's disease. This talk represents joint work with Glen Satten of the Centers for Disease Control and Prevention.

I. Măndoiu, R. Sunderraman, and A. Zelikovsky (Eds.): ISBRA 2008, LNBI 4983, p. 183, 2008.

Incorporating Literature Knowledge in Bayesian Network for Inferring Gene Networks with Gene Expression Data

Eyad Almasri, Peter Larsen, Guanrao Chen, and Yang Dai

University of Illinois at Chicago
851 S. Morgan St. (M/C 063), Chicago, IL 60607, USA
{ealmas1,plarsen,gchen4,yangdai}@uic.edu

Abstract. The reconstruction of gene networks from microarray gene expression has been a challenging problem in bioinformatics. Various methods have been proposed for this problem. The incorporation of various genomic and proteomic data has been shown to enhance the learning ability in the Bayesian Network (BN) approach. However, the knowledge embedded in the large body of published literature has not been utilized in a systematic way. In this work, prior knowledge on gene interaction was derived based on the statistical analysis of published interactions between pairs of genes or gene products. This information was used (1) to construct a structure prior and (2) to reduce the search space in the BN algorithm. The performance of the two approaches was evaluated and compared with the BN method without prior knowledge on two time course microarray gene expression data related to the yeast cell cycle. The results indicate that the proposed algorithms can identify edges in learned networks with higher biological relevance. Furthermore, the method using literature knowledge for the reduction of the search space outperformed the method using a structure prior in the BN framework.

Keywords: Bayesian Network, Likelihood score, Prior probability.

1 Introduction

The Bayesian Network (BN) has been proven to be useful and important in biomedical applications such as clinical decision support systems, information retrieval, and discovery of gene regulatory networks [1]. Automatic learning of BNs from observational data has been an area of intense research for more than a decade, yielding practical algorithms and tools [2]. The main approach for learning BNs from data is based on the strategy of search-and-score, which attempts to identify the most probable *a posteriori* network S given the data D and prior knowledge ξ. Depending on assumptions, maximizing this probability $P(S|D,\xi)$ corresponds to maximizing a score function [3]. Algorithms in this category search the space of all possible networks for the one that maximizes the score based on greedy, local, or other search strategies. The early work in the reconstruction of gene networks has been dependent on the use of microarray

I. Măndoiu, R. Sunderraman, and A. Zelikovsky (Eds.): ISBRA 2008, LNBI 4983, pp. 184–195, 2008.

data alone, largely ignoring existing prior biological knowledge [1], [4]-[8]. More recent work has shown that prior knowledge such as transcription factor binding data can be complementary to microarray data in a BN framework [9]-[12]. In the work of Hartemink et al. [9], the transcription factor binding information is incorporated into a structural prior to model the yeast galactose metabolism and pheromone response pathways [7],[9]. Tamada et al. [11] integrate the learning of transcription factor binding sites along with the learning of the genetic network. In such a framework, it is difficult to isolate the quantitative effects of increasing amounts of prior knowledge on learning performance, as the motif finder cannot be forced to learn a specific amount of prior knowledge. In a closely related work [12], a list of protein-protein interactions is mined and fed into the structural prior. As this prior knowledge is of a very specific type, the biological implications of protein-protein interactions are exploited in the learning scheme by adding nodes representing protein complexes. Other recent work allows the integration of multiple types of prior knowledge into a BN framework [13]. Werhli et al. [14] used Bayesian approach to systematically integrate expression data with multiple sources of prior knowledge. The previous research mentioned above mostly focused on the integration of high-throughput experimental data in the BN approach. The existing large body of published literature, however, is ignored. In this study, BN algorithms using a structure prior obtained from previously published literature are proposed for the inference of gene regulatory networks. In order to construct a structure prior, we use the likelihood of interaction (LOI), presented in our previous work [15], for each pair of genes or gene products based on a statistical analysis of published interactions with the Gene Ontology molecular function annotations for the interacting partners in a specific organism. The effective incorporation of this prior knowledge is then investigated through two ways. One is using the method of the search-and-score by imposing the structure prior in the learning algorithm. The other approach is using the prior knowledge to restrict the search space, which is similar to the constraint-based algorithm [16]. The difference is that in the constraint-based approach the constraints are derived from data while the constraints in our algorithm are derived from the published literature. The algorithms based on these two approaches are evaluated with two microarray datasets and compared with the BN algorithm without using prior knowledge.

2 Dataset

In this study, two subsets of microarray gene expressions related to cell-cycle dependent genes in the budding yeast *Saccharomyces Cerevisiae* microarray experiments [17] were used for the validation of the algorithms. These microarray experiments were designed to create a comprehensive list of yeast genes whose transcription levels were expressed periodically within the cell cycle. The first subset is the time course expression profiles of 102 genes that include 10 known transcription regulators and their possible regulation targets [18]. The second subset is comprised of 999 expression profiles of the most cyclically regulated

genes in the microarray experiments. The gene expressions of cell cycle synchronized yeast cultures were collected over 18 time points taken in 7-minute intervals. This time series covers more than two complete cycles of the cell division. It is highly enriched for known interacting genes involved in the *Saccharomyces* cell cycle. For this study, the true interactions were derived from the database of PathwayAssist [19] by submitting the list of genes and querying for instances of published interactions between these genes limited to interaction types expression and regulation. PathwayAssist is a bioinformatics tool that identifies possible interactions between gene products through a natural language search algorithm of all available PubMed published abstracts. Given an input set of query genes or gene products, PathwayAssist searches the database of published abstracts, seeking instances in which genes are identified as interacting according to the information found in available PubMed abstracts. The nature of interactions (expression, regulation, genetic interaction, binding, protein modification, and chemical modification as defined in that software package) can be used to screen for specific types of interactions. The software returns the set of interactions with the PubMed references from which those interactions were identified. The 102-gene set has 171 true interactions and the 999-gene set contains 729 true interactions. Gene expression profiles in both datasets were discretized for the BN analysis. For each gene, an average of expression values across all time points was calculated. Each time point was assigned a binary value according to whether the expression value at that time point was above or below this average expression value.

3 The Likelihood of Interaction (LOI) Scores

This study utilizes the concept of Likelihood of Interaction (LOI) scores for gene interaction pairs developed in our previous study [15]. The LOI-score is a measure of the likelihood that a gene or a gene product with a particular molecular function influences the expression of another gene or a gene product. This likelihood is derived from the analysis of published gene interactions and their molecular functions. More specifically, if two genes closely resemble by their molecular functions from previously observed interaction pairs, then they will be considered likely to interact. For the derivation of LOI-scores, a set of 2457 yeast genes was selected from the Saccharomyces cerevisiae database of PathwayAssist 3.0 and used to identify directed gene pairs of interaction types Expression, Regulation, and Protein Modification as defined in that software package [19]. These gene interactions are suggested by 4,192 observed interactions from the automated PubMed literature search. The 23 GO Molecular Function (MF) annotations specified by the Saccharomyces Genome Database SGD GO Slim Mapper [20] were considered for the annotation of the regulator and the target genes. The details of deriving the LOI score for each pair of GO annotations can be found in [15]. A negative LOI-score indicates that a particular GO MF annotation pair occurs less frequently than expected by random chance. A positive LOI-score indicates an interaction between GO MF annotations occurs more frequently

than expected at random. A score near zero indicates that the frequency occurs at a level near that expected by random. The calculated LOI-scores for GO MF annotation pairs was used to generate the LOI-scores for all possible gene interaction pairs of in the subsets of the yeast cell cycle microarray data. The 23 GO MF annotations described previously were applied to the genes in the subsets. For a possible interaction pair between two genes, their annotations were used for the assignment of a LOI-score for the likelihood of that interaction from the previously calculated table of LOI-scores. If a gene possesses multiple annotations, then a LOI-score was averaged between all possible pairs of annotations for a given potential interaction pair. More details on the LOI-method can be found in [15].

4 Bayesian Network Learning Algorithms

A BN is a directed acyclic graph, in which the nodes correspond to genes or their products and the edges correspond to direct probabilistic dependencies, such as causality, mediation, activation, and inhibition between the genes. Given microarray gene expression data D, the BN method discovers a network S such that the *posterior* probability $P(S|D)$ is maximized. This *posterior* is proportional to the likelihood $P(D|S)$, i.e., $P(S|D) \propto P(D|S)$ if there is no prior assumed. When the prior knowledge ξ is applied, then the *posterior* is proportional to the product of the likelihood and prior knowledge on the network, i.e.,$P(S|D,\xi) \propto P(S|\xi)P(D|S,\xi)$. The term $P(S|\xi)$ denotes the prior probability of the network S. The main approach developed for the search of highly scored networks in BNs is to search in the space of direct acyclic graphs (DAGs) [1],[22]. This task is carried out through operations including edge reversal, edge addition and edge deletion on a randomly generated network structure. The K2 score [3] was often used to evaluate the networks generated. For a given network S, this score is defined as the likelihood

$$P(D|S) = \prod_{i=1}^{n} \prod_{j=1}^{q_i} \frac{(r_i - 1)!}{(N_{ij} + r_i - 1)!} \prod_{k=1}^{r_i} N_{ijk}!, \tag{1}$$

where N_{ijk} is the number of cases in D in which variable ξ has the k^{th} value, q_i is the number of parents for i, and r_i the number of possible values of variable ξ. Thus,

$$N_{ij} = \sum_{k=1}^{r_i} N_{ijk}. \tag{2}$$

When the prior probability is considered, the K2 score can be modified as follows for a network S.

$$P(S|\xi)P(D|S,\xi) = P(S|\xi) \prod_{i=1}^{n} \prod_{j=1}^{q_i} \frac{(r_i - 1)!}{(N_{ij} + r_i - 1)!} \prod_{k=1}^{r_i} N_{ijk}!. \tag{3}$$

The aim of building a prior from the background knowledge is to bias the search for a BN toward a model that contains the preference expressed in this prior. Whenever there is not much evidence in the data against the experts' beliefs, the search will not be biased.

4.1 The Algorithm BN-NP: Using No Prior Knowledge

This algorithm consists of the following steps. First, generate a fixed number n_T of networks where each edge is included with an equal probability p ($p = 0.5$). Next, compute the K2 score for each of the n_T networks using formula (1). Select the one with the highest score and store the corresponding network S^*. Then generate n_T new random networks based on the probability determined as follows. Each edge in the network S^* is selected with a probability p_1; an edge not in S^* is selected with probability $p_0 = 1 - p_1$. Then compute the K2 score for each of the n_T networks. Select the one with the highest score. If this score is higher than the previous best score, then store the corresponding network. Repeat this process m times. The parameter p_1 determines how much confidence in each edge will go in the next iteration and was empirically determined at 0.8; we chose $m = 50$ for the 102-gene set and $m = 10$ for the 999-gene set.

4.2 The Algorithm BN-P: Using Prior Probability Derived from LOI-Scores

The structure prior was constructed from the partial knowledge on the underlying network structure. We specified our confidence in possible connections between gene pairs based on the p-values of the LOI-scores. If the p-value of an LOI-score is significant, then the corresponding interaction is believed to be more likely. Conversely, if the p-value of an LOI-score is insignificant, then our belief of the corresponding gene pairs to be interacting should be lower. The detailed assignment of prior probability for gene pairs is described as follows.

A p-value (p_{ij}) is calculated for the LOI-score of a gene interaction pair (ij) assuming normal distribution of the LOI-score. The structure prior for the edge from node i to node j is then assigned as:

$$\pi_{ij} = p(i \to j) = 1 - p_{ij}, p(i \cdots j) = 1 - \pi_{ij} = p_{ij}, \tag{4}$$

where $i \to j$ and $i \cdots j$ mean that there is an edge or no edge from node i to node j, respectively. Let e_{ij} denote the random variable which takes value 1 if there is an edge from node i to node j and takes value 0 if there is no edge between the node i and j. Then from the Bernoulli distribution the probability for random variable e_{ij} is:

$$p(e_{ij}|\xi) = \pi_{ij}^{e_{ij}}(1 - \pi_{ij})^{1-(e_{ij})}. \tag{5}$$

The structure prior constructed in this way is only an informal prior. A formal prior for the BN structure $S = (E, V)$, where E is the set of edges and V is the

set of vertices such that an acyclic digraph, can be written as follows:

$$P(S|\xi) = c \prod_{i,j \varepsilon V, i \neq j} P(e_{ij}|\xi), \tag{6}$$

where c is a normalizing constant. The normalizing constant c can be fixed at 1, as the actual magnitude of c does not affect structure searching [23]. In this algorithm, formula (3) is used to compute the K2 score and the n_T networks are generated based on the prior probability assigned to the edges using formula (6). The procedure for generating networks after the initial step is the same as that in the algorithm BN-NP.

4.3 The Algorithm BN-RP: Using Restricted Prior Knowledge

Both algorithms BN-NP and BN-P require the consideration of all nodes as potential candidates of a given node, which leads to expensive computational time. In the algorithm BN-RP, we restrict the possible candidates to those edges with significant LOI-score p-values. The set of these edges is called the LOI-score restricted set. The algorithm BN-RP samples primarily edges from this restricted set for network generation. However, it also samples edges not in this set, that is, samples edges that have insignificant p-values for LOI-scores with a smaller probability. The algorithm can be described as follows. First, generate n_T random networks by sampling edges from the LOI-score restricted set with probability p_1, and sampling edges not in the LOI-score restricted set with a small probability $p_0 = 0.2$. Then compute the K2 score with formula (1) for each of the n_T networks. Select the one with the highest K2 score and store the corresponding network S^*. Next, generate n_T new random networks based on the probability determined as follows. An edge in the network S^* is selected with a probability p_1; an edge not in the network S^* but in the LOI-score restricted set is selected with probability $(1-p_1)p_2$; and an edge not in either of the above categories is selected by probability $(1-p_1)(1-p_2)$. Then compute the K2 score with formula (1) for each of the n_T networks and select the one with the highest K2 score. Store the corresponding structure S^* if its score is higher than the previous best score. Repeat the process m times. The parameters used here are p_0=0.2; p_1=0.8; p_2=0.6. The p-value thresholds for LOI-scores were 5.12e-17 for the 102-gene set and 3e-5 for the 999-gene set, respectively.

5 Results

The performances of the three BN algorithms described previously were compared using two datasets mentioned earlier. We set $n_T = 50$ for the 102-gene set and $n_T = 10$ for the 999-gene set. The results are summarized in Table 1. The criteria used for performance evaluation include recall, percentage of included edges (%Incl. edges) and precision (Prec.). Recall is defined as the ratio of the number of true interactions in the learned network to the number of total published interactions. The percentage of included edges is defined as percentage

of the total interactions found in the learned network to the total edges in a complete graph, in which there is an edge between every pair of nodes. Precision is defined as the ratio of the number of previously published edges to the number of total interactions in the learned network. Note that within this particular cell-cycle experiment, not all truths can be found, because the literature is the aggregation of biological findings over different experimental conditions. Therefore, recall and precision reported in the results should not be interpreted in the same way when evaluating the performance of a learning method, although the definitions are the same. To evaluate the novel interactions discovered by the BN methods, percent of interaction pairs that share at least one GO Biological Process (BP) annotation (%Sharing GO-BP Annotation) was also considered. Note that GO-BP annotations are not used in derivation of prior knowledge. The GO-BP annotations were selected the SGD GO Slim Mapper [21].

For the 102-gene dataset, each algorithm was repeated 50 times and for the 999-gene dataset, each algorithm was repeated 10 times due to the lengthy computation time. The average and the standard deviation (in parentheses) of the outcomes are reported in Table 1. For the 102-gene dataset, each algorithm was repeated 50 times and the average and the standard deviation (in parentheses) of the 50 outcomes are reported in Table 1. The result with the algorithm BN-NP has a lower accuracy (18.11%) and higher percentage (1.57%) of included edges in comparison with those of (22.50%) and (1.54%) respectively obtained with BN-P. The algorithm BN-RP reached the highest accuracy (25.70%) and lowest percentage (1.46%) of included edges with this dataset. In terms of precision, the algorithm BN-RP generated the highest (29.02%) compared to those of (19.10%)

Table 1. Performance of the three algorithms

Methods	% Incl. Edges	Recall %	Prec %	% Sharing GO-BP Annotation	% Incl. Edges	Recall %	Prec %	% Sharing GO-BP Annotation
	102-gene dataset				**999-gene dataset**			
BN-NP	1.57 (0.055)	18.11 (0.004)	19.10 (0.012)	27.51 (0.062)	1.10 (0.0009)	22.81 (0.0043)	1.50 (0.0012)	9.01 (0.0016)
BN-P	1.54 (0.036)	22.50 (0.004)	24.61 (0.016)	28.20 (0.052)	1.04 (0.0004)	25.92 (0.0050)	1.80 (0.0006)	9.39 (0.0003)
BN-RP	1.46 (0.048)	25.70 (0.003)	29.02 (0.011)	29.92 (0.072)	1.02 (0.0002)	26.20 (0.0140)	1.91 (0.0013)	9.46 (0.0022)

and (24.61%) for the algorithms BN-NP and BN-P respectively. Furthermore, the shared GO-BP annotations with the algorithm BN-RP also is the highest (29.92%) in comparison with (27.51%) and (28.20%) obtained from BN-NP and BN-P respectively. From these comparisons it appears that the algorithm BN-RP performs much better than the other two methods.

For the 999-gene dataset, the average and the standard deviation (in parentheses) of the 10 outcomes are summarized in Table 1. The results indicate a similar behavior of the algorithms in comparison to that with the 102-gene set. The algorithm BN-RP has the lowest percentage (1.02%) of included edges compared with (1.10%) and (1.04%) obtained from the algorithms and BN-NP and BN-P respectively. The recall achieved by the algorithm BN-RP is (26.20%) which is higher than (22.81%) and (25.92%) by the algorithms BN-NP and BN-P respectively. Similarly, the algorithm BN-RP has the highest precision of (1.91%) compared to (1.50%) and (1.80%) obtained from the algorithms BN-NP and BN-P respectively. The reduction in the percentage of the total included edges in the case of BN-RP compared with the case of BN-NP is about (0.08%). This percentage looks small, however, the reduced number of included edges is 79,760, which is substantial. The percents of interactions sharing at least one GO-BP annotation between the regulator and the target in the learning networks are also summarized in Table 1. The pattern observed in the results with the 102-gene dataset is also in presence in those with the 999-gene dataset. The percent is (9.01%) with the algorithm BN-NP, increased to (9.39%) with the algorithm BN-P, and further elevated to (9.46%) with the algorithm BN-RP. From these comparisons it appears that the algorithm BN-RP performs much better than the other two methods by these four criteria. Another benefit in using the algorithm BN-RP is the reduction in computational time. This is because that the algorithm BN-RP searches mainly among the specified nodes for potential parents based on the threshold for statistical significance for the LOI-scores, while the other two algorithms search among the entire set of variables for possible parents.

Consistency of generated networks. Since each run of the algorithm generates a different network, it is necessary to examine how different these networks are for repeated calculations. Naturally, if the data support a causal relationship strongly, it is expected that the corresponding edge is more likely to appear in the result of multiple runs. Therefore, an interaction will be considered in the final network if it is observed in more than 30 out of the 50 networks. The algorithm BN-RP again outperforms the other two algorithms in terms of the criteria used Table 2.

we also compared our methods with the LOI-method proposed in [15] for the 102-gene set. Even though using LOI-score produces higher recall (83.04%) and higher percentage of sharing GO-BP (36.21%), the percentage of included edges for the network is also higher (21.70%), which may contain many false positives. The precision (6.30%) is the lowest compared to the results from the three BN algorithms.

Biological interpretation. The final interaction network generated by the algorithm BN-RP based on the procedure defined in the previous section was further analyzed for its potential biological significance. The regulation of chromosome structure relative to progression through cell cycle is highlighted in the subnetwork in figure 1, in which all of the gene nodes downstream of histone genes HHT1, HTA1, HTA2, HTB1, and HTB2 in the network can be found. Histones

Table 2. Results for the final network obtained from the 50 runs on the 102 gene dataset

Methods	% Included Edges	Recall %	Prec %	% Sharing GO-BP Annotation
BN-NP	1.60	18.8	19.51	26.51
BN-P	1.53	22.4	24.30	27.72
BN-RP	1.51	23.1	25.39	28.65

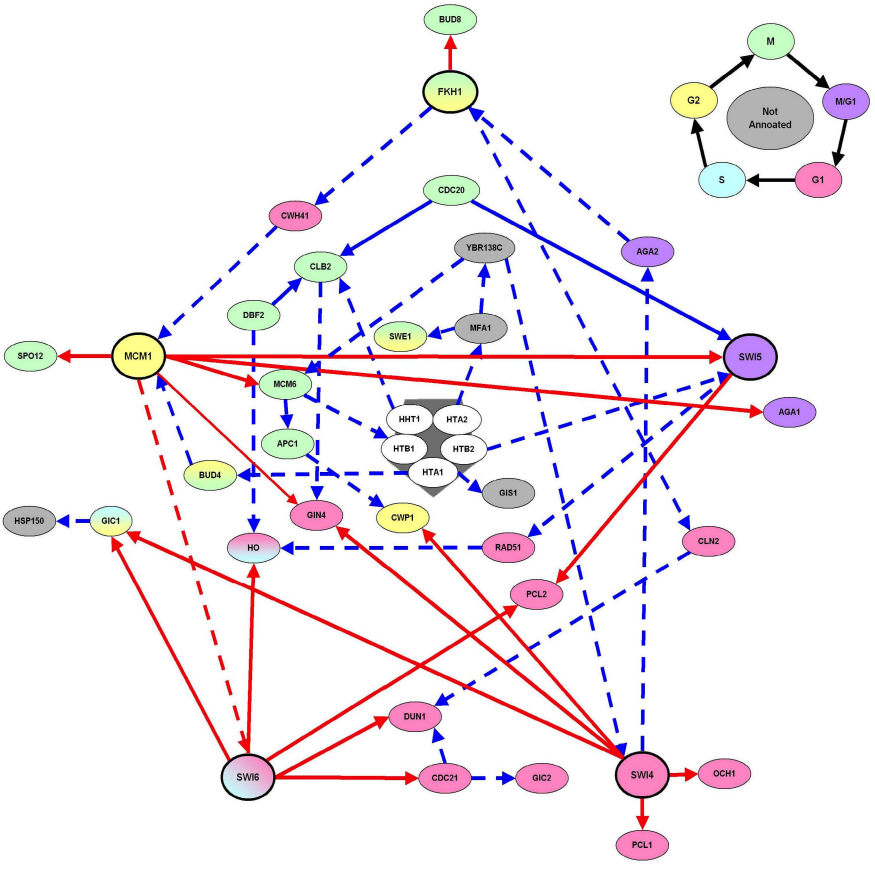


Fig. 1. Histone regulation In this figure, the subnetwork downstream of histones is identified. Nodes are colored according to their annotations of cell cycle stage specificity, except for the histone genes at the center of the figure which are white. Red lines are regulatory interactions downstream of a node annotated as a transcription factor. All other regulatory interactions are blue. Solid lines are those interactions identified in PathwayStudio as previously published, dashed lines are interactions not identified as previously published.

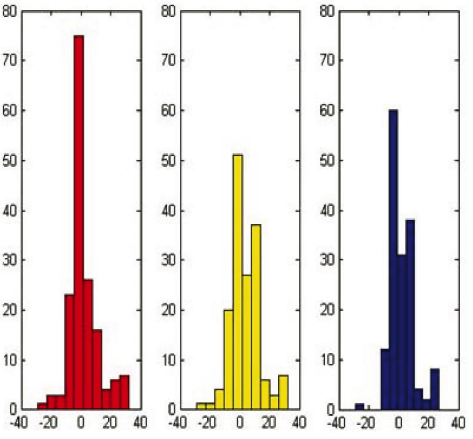


Fig. 2. Histograms of LOI-score for edges appeared in the final networks. BN-NP (red), BN-P (yellow) and BN-RP (blue).

are proteins involved in chromatin assembly or disassembly (GO:0006333). In the figure 1, nodes are highlighted according to the specific cell cycle stage to which they have been associated [17]. Histones in this network are shown to regulate key genes in the cell cycle transcriptional program. Though it is unlikely that histones directly regulate expression, it is easy to hypothesize that activity of histones can make genes available for transcription [24]-[27]. As described, the histones are shown to regulate the transcription factor SWI5. The histones are also critical for the packing of chromosomes prior to entering mitosis, as might be the case for HTA1 regulating BUD4 at the juncture of G2 and M stages. The histone HHT1 is identified to regulate G2 to M phase gene CLB2 along with previously published observations for regulations by mitosis phase regulating CBF2 and CDC20. This suggests that a change in chromatin structure is similarly crucial to CLB2s regulation of mitosis as the more direct gene expression regulation interactions. Although our method cannot identify cycles, considering the regulation of histone genes as a functional unit allows the network to be arranged in a regulatory system that successfully connects all stages of the cell cycle.

Finally, an inspection was also made for the final networks obtained from BN-P and BN-RP. Ten out of the 26 common edges are previously published interactions, indicating the similarity between BN-P and BN-RP. This feature can be further confirmed from the histograms of the LOI-scores of edges appeared in the final networks Figure 2. The algorithm BN-RP identified more edges with higher LOI-scores. Considering that structure prior could be ignored if the data support strongly certain evidence [28], it may be reasonable to conclude that the prior knowledge derived from published the literature enhances the learning ability of BN for reconstruction of gene networks from gene expressions. The algorithm BN-RP seems to be more capable in identifying interactions with biological relevance than the algorithm BN-NP.

6 Conclusion

A structure prior has been derived from the published literature for the use in the Bayesian network approach for the inference of gene networks using microarray data. Two ways for the incorporation of the prior knowledge have been investigated. Compared with not using any prior knowledge, the proposed algorithms demonstrated enhanced capability in recovering the underlying network structure. Furthermore the proposed algorithm in this study is expected to be more efficient in the reconstruction of network due to its simplicity and efficiency.

References

1. Friedman, N., et al.: Using Bayesian networks to analyze expression data. J. Computational Biology, 601–620 (2000)
2. Spirtes, P., et al.: Causation, prediction, and search. Springer, New York (1993)
3. Cooper, G., et al.: A Bayesian method for the induction of probabilistic networks from data. J. Machine Learning, 309–347 (1992)
4. Akutsu, T., et al.: Identification of genetic networks from a small number of gene expression patterns under the Boolean network model. In: Pacific Symp. Biocomputing, pp. 17–28 (1999)
5. Friedman, N., et al.: On the sample complexity of learning Bayesian networks. In: Proc. Twelfth Conference on Uncertainty in Artificial Intelligence (2001)
6. Pe'er, D., et al.: Inferring subnetworks from perturbed expression profiles. Bioinformatics, 215–S224 (2001)
7. Hartemink, A., et al.: Using graphical models and genomic expression data to statistically validate models of genetic regulatory networks. In: Pacific Symp. Biocomputing, pp. 422–433 (2002)
8. Imoto, S., et al.: Estimation of genetic networks and functional structures between genes by using Bayesian networks and nonparametric regression. In: Pacific Symp. Biocomputing, pp. 175–186 (2002)
9. Hartemink, A., et al.: Combining location and expression data for principled discovery of genetic regulatory network models. In: Pacific Symp. Biocomputing, pp. 437–449 (2002)
10. Chrisman, L., et al.: Incorporating biological knowledge into evaluation of causal regulatory hypotheses. In: Pacific Symp. Biocomputing, pp. 128–139 (2003)
11. Tamada, Y., et al.: Estimating gene networks from gene expression data by combining Bayesian network model with promoter element detection. Bioinformatics, 227–236 (2003)
12. Nariai, N., et al.: Using protein-protein interactions for refining gene networks estimated from microarray data by Bayesian networks. In: Pacific Symp. Biocomputing, pp. 336–347 (2004)
13. Yeang, C., et al.: Physical network models. J. of Computational Biology, 243–262 (2004)
14. Werhli, A., et al.: Reconstructing gene regulatory networks with bayesian networks by combining expression data with multiple sources of prior knowledge. Statistical Applications in Genetics and Molecular Biology 6(1), Article 15 (2007)
15. Larsen, P., et al.: A statistical method to incorporate biological knowledge for generating testable novel gene regulatory interactions from microarray experiments. BMC Bioinformatics 317 (2007)

16. Spirtes, P., et al.: Causation, prediction, and search. The MIT Press, New York (2000)
17. Spellman, P., et al.: Comprehensive identification of cell cycle-regulated genes of the yeast Saccharomyces cerevisiae by microarray hybridization. Molecular Cell, 3273–3297 (1998)
18. Zou, M., et al.: A new dynamic Bayesian network (DBN) approach for identifying gene regulatory networks from time course microarray data. Bioinformatic, 71–79 (2005)
19. Nikitin, A., et al.: Pathway studio–the analysis and navigation of molecular networks. Bioinformatic, 2155–2157 (2003)
20. `http://db.yeastgenome.org/cgi-bin/GO/goTermMapper`
21. Battle, A., et al.: Probabilistic discovery of overlapping cellular processes and their regulation. In: Proc. of the Annual International Conference on Computational Molecular Biology, pp. 167–176 (2004)
22. Friedman, N., et al.: Being Bayesian about network structure: A Bayesian approach to structure discovery in Bayesian networks. J. Machine Learning, 601–620 (2003)
23. Castelo, R., et al.: Priors on network structures. Biasing the search for Bayesian networks, Technical Report (CWI) (Centre for Mathematics and Computer Science) (1998)
24. Norris, D., et al.: The effect of histone gene deletions on chromatin structure in Saccharomyces Cerevisiae. Science, 759–761 (1988)
25. Luger, K., et al.: Crystal structure of the nucleosome core particle at 2.8 A resolution. Nature, 251–260 (1997)
26. Briggs, S., et al.: Gene silencing: trans-histone regulatory pathway in chromatin. Nature, 498 (1997)
27. Krogan, N., et al.: The Paf1 complex is required for histone H3 methylation by COMPASS and Dot1p: linking transcriptional elongation to histone methylation. Molecular Cell, 721–729 (2003)
28. Castelo, R., et al.: Priors on network structures. Biasing the search for Bayesian networks. International Journal of Approximate Reasoning, 39–57 (2000)

Integrative Network Component Analysis for Regulatory Network Reconstruction

Chen Wang[1], Jianhua Xuan[1,*], Li Chen[1], Po Zhao[2], Yue Wang[1], Robert Clarke[3], and Eric P. Hoffman[2]

[1] Department of Electrical and Computer Engineering, Virginia Polytechnic Institute and State University, Arlington, VA 22203, USA
{topsoil, xuan, lchen06, yuewang}@vt.edu
[2] Research Center for Genetic Medicine, Children's National Medical Center, Washington, DC 20010, USA
ehoffman@cnmcresearch.org
[3] Departments of Oncology and Physiology & Biophysics, Georgetown University School of Medicine, Washington, DC 20057, USA
clarker@georgetown.edu

Abstract. Network Component Analysis (NCA) has shown its effectiveness in regulator identification by inferring the transcription factor activity (TFA) when both microarray data and ChIP-on-chip data are available. However, the NCA scheme is not applicable to many biological studies due to the lack of complete ChIP-on-chip data. In this paper, we propose an integrative NCA (iNCA) approach to combine motif information, limited ChIP-on-chip data, and gene expression data for regulatory network inference. Specifically, a Bayesian framework is adopted to develop a novel strategy, namely stability analysis with topological sampling, to infer key TFAs and their downstream gene targets. The iNCA approach with stability analysis reduces the computational cost by avoiding a direct estimation of the high-dimensional distribution in a traditional Bayesian approach. Stability indices are designed to measure the goodness of the estimated TFAs and their connectivity strengths. The approach can also be used to evaluate the confidence level of different data sources, considering the inevitable inconsistency among the data sources. The iNCA approach has been applied to a time course microarray data set of muscle regeneration. The experimental results show that iNCA can effectively integrate motif information, ChIP-on-chip data and microarray data to identify key regulators and their gene targets in muscle regeneration. In particular, several identified TFAs like those of MyoD, myogenin and YY1 are well supported by biological experiments.

Keywords: Network component analysis, gene regulatory networks, microarray data analysis, ChIP-on-chip, muscle regeneration.

1 Introduction

With recent advances in biotechnology, genome-wide data in different levels provide complementary information about molecular mechanisms underlying various diseases.

* Corresponding author.

I. Măndoiu, R. Sunderraman, and A. Zelikovsky (Eds.): ISBRA 2008, LNBI 4983, pp. 196–207, 2008.

Particularly, high-throughput biological data have enabled us to study genome systems from a global perspective that may lead to a better understanding of their underlying biological processes [1]. Many computational methods have been proposed to identify gene modules, interactions and pathways in biological systems [2-5]. Among them, most methods assume that the expression activity of an entire gene population results from a much smaller number of latent factors such as transcription factors. This assumption not only coincides with the modular view of biological systems, but it also makes the computational task much easier [2]. For gene regulatory network modeling, there are two major trends in the literature: the first trend is to use clustering methods to explore the similarity in expression patterns [2], whereas the second trend uses decomposition methods to infer latent (hidden) factor activities [3-5].

It is often difficult to interpret the results from pure computational approaches due to the lack of supporting biological knowledge support. Biological regulatory systems are complex in nature, and key activities may occur simultaneously in the genome, transcriptome and proteome. Hence, any computational model based only on mRNA measurements may be too simple to describe the entire system. Recently, many researchers have tried to integrate multiple data sources to infer and reconstruct biological networks. For example, network component analysis (NCA) is a topological knowledge based algorithm that utilizes both protein binding data and gene expression data to reveal underlying transcription factor activities [6]. NCA has been shown to be effective in finding cell cycle regulators in yeast. Despite its success in yeast data, some issues prevent NCA to infer regulatory networks other than in yeast. First, complete biological connection data, such as high-throughput ChIP-on-chip data, are often not (or only partially) available for common species including rodent and human. Second, when different heterogeneous data sources are integrated for computational inference, the consistency of different data sources is often not guaranteed. Third, since topological knowledge (network connections) also comes from biological experiments, this knowledge likely also contains many false-positives/negatives that can lead to incorrect network inference.

In this paper, we propose a Bayesian principled integrative NCA (iNCA) approach for regulatory network inference. First, the topology information is augmented by combining motif information, ChIP-on-chip data and relevant published data. Second, with the awareness of false-positives contained in any given information, calculated priori or empirical priori can be given in order to validate the reliability of the final estimation. Third, our scheme has remarkable computational efficiency, when compared with its Bayesian counterparts [7, 8], in algorithm implementation. The concept comes from topological sampling, but we avoid the burden of the high computation cost and the large data requirements of direct distribution estimation. Instead, a stability index, pair-wise correlation measurement, is designed to evaluate the goodness of the estimated TFAs. A second stability index, a confidence measure by frequency count, is utilized to rank the most significant downstream targets of the TFs. Moreover, given the average frequency count from different knowledge sources, the inconsistency among different data sources can also be evaluated.

The iNCA scheme has been applied to a muscle regeneration microarray data set for regulatory network inference. With our new scheme, not only several key transcription factors participating in regeneration process were identified, but also their activities across the time points were correctly estimated. The downstream genes of

MyoD were also identified by ranking the frequency count; the higher ranked gene group did show more significant relationship with MyoD. Finally, the averaged frequency count score clearly showed the difference between two ChIP-on-chip sources: Myoblast (undifferentiated muscle) and Myotube (differentiated muscle), and the difference was consistent with the biological understanding of the muscle regeneration process.

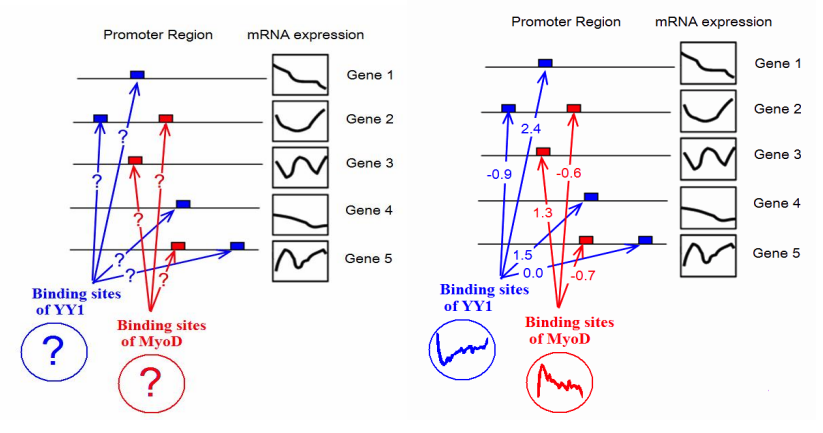
2 Method

2.1 Network Component Analysis (NCA)

Network Component Analysis (NCA) is a computational method to infer latent factors and the connection relationship of a network, given the initial topology (connection) information and the measurement of gene expression. In Fig. 1, we illustrate the NCA approach with an example from muscle regeneration studies [9]. The mathematical model of NCA can be formulated as

$$\begin{aligned} \mathbf{E}_{N\times M} &= A_{N\times L}\mathbf{T}_{L\times M}, \\ s.t.\ A &\in Z_0, \end{aligned} \tag{1}$$

where **E** is the observation, A connection matrix, **T** latent factors, and Z_0 the initial topology of the network. L is the number of latent (hidden) factors, M the number of experiment conditions, and N the number of genes.



(a) (b)

Fig. 1. An illustrative example for the NCA approach as in muscle regeneration studies. The network topology is formed by the connection matrices of the transcription factors (TFs) such as YY1 and MyoD to their target genes as shown in (a). The main objective of the NCA approach is to estimate the transcription factors' activities (TFAs) and their target genes via the estimated connection matrices as shown in (b).

As illustrated in Fig. 1, the latent factors are the transcription factors such as YY1 and MyoD; the network topology is formed by the connection matrices of the TFs to their target genes. The main objective of the NCA approach is to estimate the transcription factors' activities (TFAs) and their target genes. The NCA optimization criterion can be simply denoted as [6]:

$$\begin{aligned} &\min \| \mathbf{E}_{N\times M} - A_{N\times L}\mathbf{T}_{L\times M} \|^2, \\ &s.t.\ A \in Z_0. \end{aligned} \tag{2}$$

The NCA algorithm was originally developed for gene regulatory network reconstruction. The model (1) can be interpreted in this way: the N genes' expression pattern under M different conditions can be seen as a combination effect of L transcription factors (TFs). Note that it is well accepted that a linear model only holds after log-ratio transform [6]:

$$\log(\mathbf{E}r_{N\times M}) = A_{N\times L}\log(\mathbf{T}r_{L\times M}), \tag{3}$$

where $\mathbf{E}r_{ij} = E_{ij}(t)/E_{ij}(0)\ (i=1,...,M;\ j=1,...,N)$ and $\mathbf{T}r_{kl} = T_{kl}(t)/T_{kl}(0)$ $(k=1,...,L;\ l=1,...,M)$ are ratios of gene expression values and transcription factor activities (TFAs), respectively. In the original NCA scheme, the topology information Z_0 is provided by the ChIP-on-chip data [10]. With the ChIP-on-chip data available in yeast, NCA has been successfully applied to yeast stress response and cell cycle experiments. Among the estimated TFAs with an oscillation pattern, 75% correspond to known cell-cycle regulators [11]. However, this NCA scheme is not readily applicable to many other biological studies due to the lack of topology information. In the next section, we will use motif information as a practical means to obtain the initial topology information for NCA.

2.2 Motif Analysis for Initial Topology Information

A transcription factor (TF) is a protein that regulates its target gene's transcription by binding to a specific regulatory motif in the DNA of the promoter region(s). Thus, we can utilize regulatory motif information to establish the putative topologic relationship between a TF and a downstream target gene. Below we propose a motif analysis procedure to obtain the initial topology information for network reconstruction.

First, the upstream regions of the genes can be extracted from the database PromoSer [12]. Second, Match™ [13] (or its improved version, P-Match [14]) can be used to search the transcription factor binding sites (TFBSs) in each upstream region; this approach generates the scores of both "core similarity" and "matrix similarity" for each matched motif. Third, Match™ searches the TFBS using its position-weighted matrices (PWMs) that can be extracted from the TRANSFAC 11.1 Professional Database [15]. Fourth, according to the PWMs, a motif score can be calculated for each TF-gene pair where the score is the maximum of the average scores of core similarity and matrix similarity. These motif scores provide the initial topology information for further iNCA analysis as is detailed in the next section.

Note that each motif is a relative short sequence pattern, thus the topology from motif information is merely a rough estimation and will usually include many false

positives/negatives. While the topology information is often unreliable for any specific TF-gene pair, we can still infer some key transcription factor activities from gene expression and DNA sequence information using the stability analysis procedure developed in the next section.

2.3 Integrative Network Component Analysis (iNCA)

If we regard the network topology knowledge (i.e., connection matrices; hereafter denoted as **K**) as priori and expression measurements (**E**) as observations, the Bayesian interpretation of the NCA learning algorithm can be applied to maximize the posterior probability as following:

$$\Pr(A, \mathbf{T} / \mathbf{E}, \mathbf{K}), \tag{4}$$

where A and **T** are defined as in Equation (1). Traditional NCA assumed that **K** is "almost" deterministic and with the identifiable conditions, a solution with posterior probability of one can be obtained [6]. However, we know that not only the knowledge itself contains false positives, but different knowledge sources are also prone to introducing topology inconsistency. Therefore, the overall posterior probability should be formulated as:

$$\Pr(A, \mathbf{T}, \mathbf{K} / \mathbf{E}) = \Pr(A, \mathbf{T} / \mathbf{E}, \mathbf{K}) \Pr(\mathbf{K}). \tag{5}$$

In Equation (5), $\Pr(\mathbf{K}) = \prod_{i=1}^{S} \Pr(K_i)$ approximates a joint priori probability of S different knowledge sources (K_i, i=1,…,S). For example, we can calculate the corresponding priori probability from sequence motif's position-weighted matrix (PWM) score, or from ChIP-on-chip data's p-value [7]. However, it is still a high-dimensional distribution estimation problem to maximize the posterior probability in Equation (5). In the past, some researchers proposed to use Gibbs sampling technique to directly estimate the distribution, which, unfortunately, suffers from a very high computation cost and biased parameter estimation when limited data samples are available [8, 11].

Here, we propose an alternative approach to avoid direct estimation of posterior probability for iNCA. The basic idea is to apply a stability analysis together with topological sampling. Stability analysis was originally proposed to perform model selection for unsupervised learning, where the number of clusters can be correctly estimated [16]. The basic idea of stability analysis is that if a small perturbation is introduced equally in different model order, the best consistency will only occur when the model fits correctly the underlying structure of the data.

Here we develop a stability analysis procedure to assess the estimation results of iNCA. Since true functional data on TFAs are usually unavailable, we must establish whether an estimated TFA is a reliable estimate or if this prediction has arisen by error or by chance. When the topology information, from motif analysis and/or ChIP-on-chip data, contains many false positives/negatives, we must also determine which TFAs are the reliable estimates of underlying transcription factor activities, or whether these are simply random outcomes.

If we intentionally perturb the network topology, each of the estimated TFAs will change. A falsely or poorly estimated TFA tends to be altered easily by small

perturbations and will appear to be unstable. On the contrary, a good TFA estimation, reflecting the consistency between microarray expression data and topology knowledge, will tend to keep its activity pattern throughout multiple perturbations. Therefore, random perturbations should be performed multiple times to test the stability of each predicted TFA.

Algorithm 1:

Sample the topology from knowledge prior distribution $\Pr(\mathbf{K})$ for P times.

For $i = 1,\ldots,P$

Maximize $\Pr(A = A_i, \mathbf{T} = \mathbf{T}_i / \mathbf{E}, \mathbf{K}) \Pr(\mathbf{K} = \mathbf{K}_i)$ to obtain $\mathbf{T}_i$ (via NCA).

End of sampling

For $j = 1,\ldots,L$ (i.e., the j^{th} TF)

Compute stability measurements of j^{th} TFA = $\{|\, CorrCoef(TFA_j(p), TFA_j(q))\,|_{p \neq q}\}$.

End of the j^{th} TF

To be concrete, under specific conditions (given the expression observations **E**), not all the regulators or TFs from the literatures play the same role; some of them play a key role relevant to the biological study, and some are not relevant at all. Intuitively, the relevant ones tend to have a consistent estimation with observations **E** when moderate perturbations introduced. In other words, moderate perturbations are unlikely to destroy the estimate. Alternatively, given the priori distribution of knowledge, the posterior distribution of a relevant TF will tend to have a narrow distribution (around the true solution). Therefore, we propose to measure the pair-wise similarity of each TF after different topological sampling to the knowledge set **K**. This stability-based algorithm is summarized in Algorithm 1.

Note that in the stability measurement, p and q correspond to different topological samplings, respectively. *CorrCoef*() is the Pearson correlation coefficient function. When stability measurements of a specific TFA are obtained, we can use several statistics including mean and variance estimates to describe a predicted TFA's robustness with respect to perturbation. In this paper, we use *boxplot* to visualize the stability measurement, simultaneously depicting its minimum, 25% percentile, median, 75% percentile, and maximum.

Similarly, we do not need to estimate the distribution of A, but to rank A according to the same TF. That is to find which genes are the most significant downstream targets of a specific TF. Although A describes the controlling strength from TFs to genes, we do not need to estimate the exact value of A. Rather, we try to find out whether this connection is the most significant connection, or equivalently, whether this connection is kept with a high frequency when different knowledge distribution samplings are applied. Hence, we propose a simple stability index (confidence measure by frequency count) to measure the connection strength in this paper. The stability-based approach is summarized in Algorithm 2.

From the approach above, we can obtain different subsets from different knowledge sources according to the same frequency count threshold. Even above the same frequency count threshold, the variability between different knowledge sets will result

Algorithm 2:
Sample the topology from knowledge prior distribution $\Pr(\mathbf{K})$ for P times.
For i= 1,…,P (in our experiments, we set P = 1000)
Maximize $\Pr(A = A_i, \mathbf{T} = \mathbf{T}_i / \mathbf{E}, \mathbf{K}) \Pr(\mathbf{K} = \mathbf{K}_i)$ to obtain A_i (via NCA).

For j = 1,…,L (i.e., the j[th] TF)

For a specific TF, set a threshold *TOP_RANK* (e.g., *TOP_RANK* = 100).
If the j-th TF's controlling strength to the l-th gene $A_i(j,l)$ is among the top *TOP_RANK* of all the $A_i(j,l)$, l=1,...,N, frequency count is incremented by one to the corresponding position:

$$Count_{j,l} = Count_{j,l} + 1 .$$

End of the j^{th} TF
End of sampling

in different average frequency counts, from which we can evaluate the quality of knowledge sources themselves.

3 Experimental Results

3.1 Data Set Description

Staged skeletal muscle degeneration/regeneration was induced by injection of cardiotoxin (CTX) as previously described [17]. Two mice were injected in gastrocnemius muscles of both sides, and then sacrificed at each of the following time points: 0, 12h(ours), 1d(ay), 2d, 3d, 3.5d, 4d, 4.5d, 5d, 5.5d, 6d, 6.5d, 7d, 7.5d, 8d, 8.5d, 9d, 9.5d, 10d, 11d, 12d, 13d, 14d, 16d, 20d, 30d, and 40d. The time course microarray data set was acquired with Affymetrix's Murine Genome U74v2 Set from an expression profiling study at CNMC. We used Affymetrix's MAS 5.0 probe set interpretation algorithm to process the original intensity data for gene expression measurements. After the processing, we obtained the expression measurements of 7570 probe sets in each sample.

3.2 Initial Topology Information

From the TRANSFAC 11.1 Professional Database, 24 mouse muscle related transcription factors were selected for motif analysis (see Fig. 2). According to their position weighted matrices (PWMs), possible connection topology was calculated. As described in the previous section, each possible connection has a motif score obtained from the TRANSFAC database. In this study, the connections from motif analysis with score above 0.98 were kept for further analysis. In addition, ChIP-on-chip experiments on a specific TF, MyoD, were done in two conditions: Myoblast (MB; undifferentiated muscle) and Myotube (MT; differentiated muscle). For MyoD's ChIP-on-chip data, the connections whose binding p-values below 0.01 were kept for

further analysis. It is worth noting that for the cells (c2c12) used for the ChIP-on-chip data, MyoD is expressed in both MB and MT. But for the *in vivo* study (the cells used for 27 time point microarray data), MyoD expression is barely detectable in MB (residing in non-injected muscles), but increasing greatly during the transition of MB to MT (muscle regeneration).

3.3 Integrative NCA and Stability Analysis

From Equation (3), we know that a log-ratio operation should be performed on the data set to ensure that the linear model holds. We chose the last (27^{th}) time point sample as the reference for calculating the ratios, because it is at the 40^{th} day, the late stage of the muscle regeneration, is considered to provide a normal muscle reference.

In this experiment, 1000 times of independent topological samplings were carried out. The priori of motif knowledge was set to 0.5, and the priori of ChIP-on-chip knowledge was set to 0.7. Thus, each motif-based connection has a 50% chance to be randomly deleted during the topological sampling, and each ChIP-on-chip based connection has a 30% chance to be randomly deleted during the sampling. The stability measurement was calculated; the resulting boxplot is shown in Fig. 2. It can be seen from Fig. 2 that some of estimated TFAs are stable during perturbation, including the TFAs of YY1, myogenin and MyoD (shown in red in Fig. 2). In the procedure, for each sampling the top 100 connections with MyoD were added with one frequency count. The confidence measure by frequency count is shown in Fig. 3 for the downstream gene targets of MyoD.

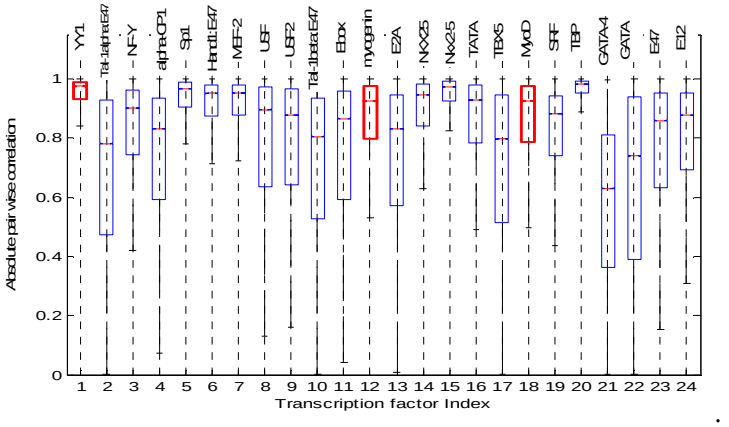


Fig. 2. Stability measurements using the first perturbation procedure. The boxes with red color are the stability measurements of YY1, myogenin and MyoD, respectively.

Because all the topology information contains false-positives and it is inappropriate to fit to specific muscle regeneration data, stability analysis is used to find those transcription factors with stable estimated activities throughout the random sampling procedure. Although there are more than ten stable TFAs from the analysis, we focus here on three: MyoD, myogenin, and YY1. From the literature, these three TFs are key regulators of muscle differentiation [17-19]. In Fig. 4, we show the expression

profiles and corresponding TFAs of these three TFs. It can be seen from Fig. 4 that these predicted TFAs are biologically relevant to muscle regeneration because the TFAs exhibit sudden increases in their log expression ratios after muscle injury and these values gradually decrease in the later stages of muscle regeneration when the tissue has almost completed regeneration.

For YY1, a large difference between its measured gene expression level and inferred TFA is evident in Fig. 4(a) and Fig. 4(b). The YY1 gene expression log-ratio is relative low when compared with other TFs, and its trend has no obvious relationship with muscle regeneration. However, the inferred TFA shows a close relationship with the regeneration process. This is supported by a biological study [19] that reported an inconsistency between YY1 protein and mRNA expression levels and showed an important role for YY1 in mouse muscle differentiation. Specifically, YY1 acts as a transcription repressor, down-regulating muscle gene expression in undifferentiated muscle cells [20]. During muscle differentiation, YY1's activity is decreased, which leads to the induction of muscle gene expression. The reduction in YY1 activity occurs at the protein rather than mRNA level. YY1 protein is degraded by a protease, calpain II (m-calpain), in differentiating muscle cells [19]. Thus, our inferred YY1 TFA from the muscle regeneration data set is well supported by the biological observation of an induction of calpain II and relatively less change of YY1 mRNA expression in muscle regeneration. It can also be observed that calpain II's mRNA expression levels have a very similar pattern with our estimated YY1 TFA, with a correlation coefficient of r>0.9.

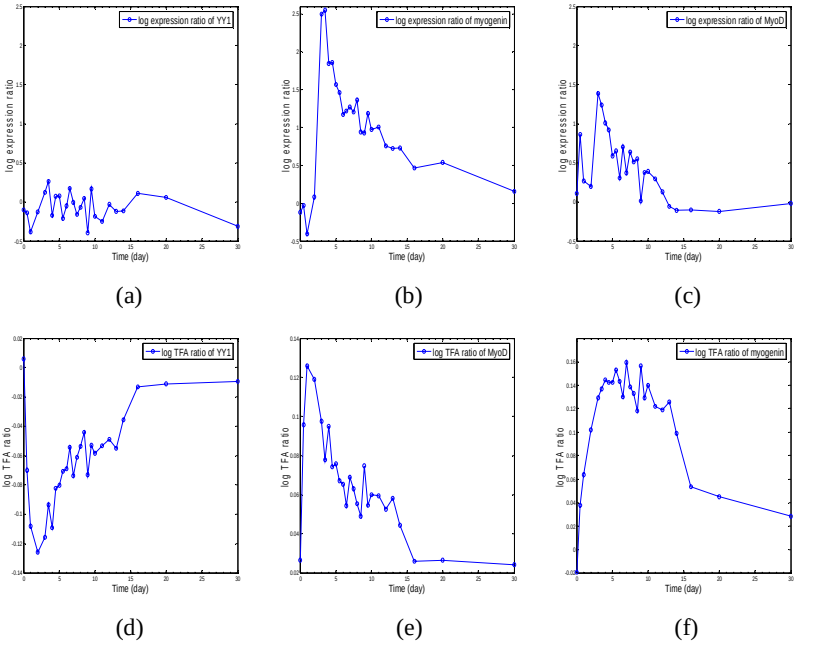


Fig. 3. Gene expression patterns of (a) YY1, (b) myogenin, and (c) MyoD, respectively; estimated TFAs of (d) YY1, (e) myogenin, and (f) MyoD, respectively. Note: x-axis – time points; y-axis – log expression ratio (a, c and e) or log TFA ratio (b, d, e).

Using the frequency count in Fig. 4(a), we fed the top 100 downstream genes into the Ingenuity Pathway Analysis (IPA; http://www.ingenuity.com). The most significant network involved with MyoD is shown in Fig. 4(b), which is highly related to muscle development and differentiation. As we can see from Fig. 4(b), there are 16 directly related genes with MyoD, and several key muscle regeneration factors (MYC, MYOG, and MEF2C) involved. This indicates that MyoD's regulatory power extends beyond its immediate downstream targets, as it may also control other TFs that propagate the signals initiated by MyoD.

Although the two different ChIP-on-chip data sets (76 probe ids from MB and 100 probe ids from MT) were treated equally in the sampling, from a biological perspective these come from very different experimental backgrounds. The MB list was obtained from undifferentiated muscle cells, and the MT list was obtained from differentiated muscle cells. The MT list should be more consistent with muscle regeneration microarray data than the MB list, since muscle regeneration is mainly involved with muscle differentiation after injury. From the frequency count obtained by topological sampling, a threshold of 100 was chosen to calculate the top probe ids' average frequency count, that is, only the downstream probe ids which have more than 100 times appearing in the top 100 were chosen. From these analyses we obtained 14 probe ids from the MB list with an average frequency count of 245.9 (out of 1000 independent samplings), and 32 probe ids from the MT list with an average frequency count of 333.4 (out of 1000 independent samplings). Therefore, the downstream targets in the MT list have a larger average frequency count than those in the MB list. The difference indicated that there are more downstream targets of MyoD in MT than MB, which seems consistent with the biological understanding of the experiments.

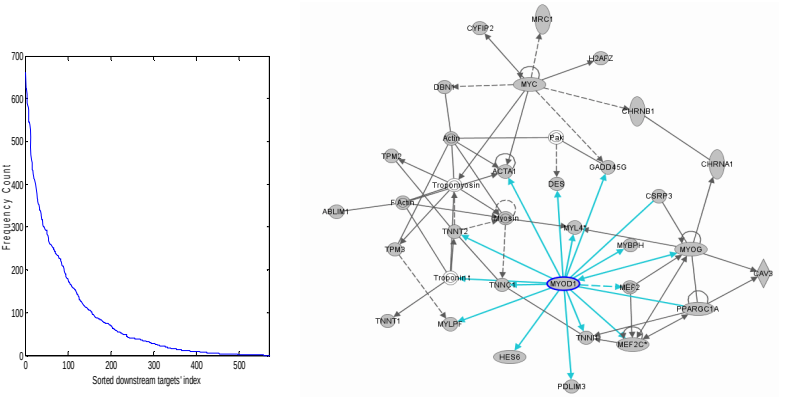


Fig. 4. (a) The sorted frequency count of MyoD's target genes; x-axis is the sorted downstream targets' index of MyoD, and y-axis is the corresponding frequency count. (b) The most significant network involved with MyoD from the Ingenuity Pathway Analysis.

4 Conclusions

In this paper, we propose a Bayesian principled, integrative network component analysis (iNCA) approach, to infer underlying regulatory activities by integrating

motif information, ChIP-on-chip and gene expression data. Since many false positives/negatives likely exist in both motif information and ChIP-on-chip data, we have further developed a stability analysis procedure for iNCA to extract stable TFAs and their downstream gene targets. To reduce the computational cost and the amount of required for traditional Bayesian approaches, we specifically designed the stability indices to measure the goodness of the estimated TFAs and their connectivity strengths. The iNCA scheme was applied to a time course microarray data set from a muscle regeneration profiling study. The experimental results show that our new approach can reveal both key regulators and their target genes, and also discover novel regulatory mechanisms potentially involved in muscle regeneration. By further incorporating biological knowledge, we hope to extend this approach to analyzing muscle dystrophy data for novel pathway discovery and biomarker identification [9].

Acknowledgments. This research was supported in part by NIH Grants (NS29525-13A, EB000830, CA109872 and CA096483) and a DoD/CDMRP grant (BC030280).

References

1. Slonim, D.K.: From patterns to pathways: gene expression data analysis comes of age. Nat. Genet. 32, 502–508 (2002)
2. Segal, E., et al.: Module networks: identifying regulatory modules and their condition-specific regulators from gene expression data. Nat. Genet. 34(2), 166–176 (2003)
3. Lee, S.I., Batzoglou, S.: Application of independent component analysis to microarrays. Genome. Biol. 4(11), 76 (2003)
4. Gong, T., et al.: Latent Variable and nICA Modeling of Pathway Gene Module Composite. In: Engineering in Medicine and Biology Society, 2006. EMBS 2006. 28th Annual International Conference of the IEEE, pp. 5872–5875 (2006)
5. Pascual-Montano, A., et al.: bioNMF: a versatile tool for non-negative matrix factorization in biology. BMC Bioinformatics 7, 366 (2006)
6. Liao, J.C., et al.: Network component analysis: reconstruction of regulatory signals in biological systems. Proc. Natl. Acad. Sci. USA 100(26), 15522–15527 (2003)
7. Chen, G., Jensen, S.T., Stoeckert Jr., C.J.: Clustering of genes into regulons using integrated modeling-COGRIM. Genome. Biol. 8(1), 4 (2007)
8. Sabatti, C., James, G.M.: Bayesian sparse hidden components analysis for transcription regulation networks. Bioinformatics 22(6), 739–746 (2006)
9. Bakay, M., et al.: Nuclear envelope dystrophies show a transcriptional fingerprint suggesting disruption of Rb-MyoD pathways in muscle regeneration. Brain 129(Pt 4), 996–1013 (2006)
10. Lee, T.I., et al.: Transcriptional regulatory networks in Saccharomyces cerevisiae. Science 298(5594), 799–804 (2002)
11. Yang, Y.L., et al.: Inferring yeast cell cycle regulators and interactions using transcription factor activities. BMC Genomics 6(1), 90 (2005)
12. Halees, A.S., Leyfer, D., Weng, Z.: PromoSer: A large-scale mammalian promoter and transcription start site identification service. Nucleic. Acids. Res. 31(13), 3554–3559 (2003)
13. Kel, A.E., et al.: MATCH: A tool for searching transcription factor binding sites in DNA sequences. Nucleic. Acids. Res. 31(13), 3576–3579 (2003)

14. Chekmenev, D.S., Haid, C., Kel, A.E.: P-Match: transcription factor binding site search by combining patterns and weight matrices. Nucleic Acids. Res. 33(Web Server issue), 432–437 (2005)
15. Matys, V., et al.: TRANSFAC and its module TRANSCompel: transcriptional gene regulation in eukaryotes. Nucleic Acids. Res. 34(Database issue), 108–110 (2006)
16. Lange, T., et al.: Stability-Based Model Selection. In: Advances in Neural Information Processing Systems (NIPS 2002) (2002)
17. Zhao, P., et al.: In vivo filtering of in vitro expression data reveals MyoD targets. C R Biol. 326(10-11), 1049–1065 (2003)
18. Blais, A., et al.: An initial blueprint for myogenic differentiation. Genes. Dev. 19(5), 553–569 (2005)
19. Walowitz, J.L., et al.: Proteolytic regulation of the zinc finger transcription factor YY1, a repressor of muscle-restricted gene expression. J. Biol. Chem. 273(12), 6656–6661 (1998)
20. Galvagni, F., et al.: The dystrophin promoter is negatively regulated by YY1 in undifferentiated muscle cells. J. Biol. Chem. 273(50), 33708–33713 (1998)

A Graph-Theoretic Method for Mining Overlapping Functional Modules in Protein Interaction Networks*

Min Li[1], Jianxin Wang[1,**], and Jianer Chen[1,2]

[1] School of Information Science and Engineering,
Central South University, Changsha 410083, P.R. China
[2] Department of Computer Science,
Texas A&M University, College Station, TX 77843, USA
limin@mail.csu.edu.cn,jxwang@mail.csu.edu.cn,chen@cs.tamu.edu
http://netlab.csu.edu.cn

Abstract. Identification of functional modules in large protein interaction networks is crucial to understand principles of cellular organization, processes and functions. As a protein can perform different functions, functional modules overlap with each other. In this paper, we presented a new algorithm OMFinder for mining overlapping functional modules in protein interaction networks by using graph split and reduction. We applied algorithm OMFinder to the core protein interaction network of budding yeast collected from DIP database. The experimental results showed that algorithm OMFinder detected many significant overlapping functional modules with various topologies. The significances of identified modules were evaluated by using functional categories from MIPS database. Most importantly, our algorithm had very low discard rate compared to other approaches of detecting overlapping modules.

Keywords: protein interaction network, functional module, graph.

1 Introduction

Proteins are central components of cell machinery and life [1]. Large-scale interaction detection methods have resulted in a large amount of protein-protein interaction data. Such data can be naturally represented in the form of networks. System level analysis and understanding of protein interaction networks is one of the most fundamental challenges in post-genomic era. Accumulating evidence suggests these protein interaction networks are organized by functional modules, which are cellular entities performing certain biological functions [2,3,4,5,6].

* This research was supported in part by the National Natural Science Foundation of China under Grant Nos. 60433020 and 60773111, the Program for New Century Excellent Talents in University No. NCET-05-0683, the Program for Changjiang Scholars and Innovative Research Team in University No. IRT0661.

** The corresponding author.

I. Măndoiu, R. Sunderraman, and A. Zelikovsky (Eds.): ISBRA 2008, LNBI 4983, pp. 208–219, 2008.

Identification of functional modules is crucial in understanding the principles of cellular organization and unveiling functional and evolutionary mechanisms.

A wide range of graph clustering algorithms have been developed to identify functional modules from protein interaction networks. All these methods can be categorized into three groups: partitional clustering, hierarchical clustering and density-based clustering.

Partitional clustering approaches partition a network into multi separated sub-networks. As a typical example, the Restricted Neighborhood Search Clustering (RNSC) algorithm [7] explores the best partition of a network using a cost function. It starts with randomly partitioning a network, and iteratively moves a node from one cluster to another to decrease the total cost of clusters. It can get the best partition by running multi-times. However, it needs the number of clusters as prior knowledge and its results depend heavily on the quality of initial clustering.

Hierarchical clustering approaches have been applied widely for identifying functional modules [6,8,9,10,11]. Hartuv and Shamir use minimum cut set to divide network recursively [8]. Girvan and Newman decompose a network based on the graph theoretical concept of betweenness centrality [9]. Luo and colleagues also use betweenness and develope an agglomerate algorithm named MoNet [6]. Several approaches have been proposed for weighting protein-protein interactions. Pereira-Leal and colleagues propose an approximate solution to weight a protein interaction based on the number of experiments that support the interaction [10]. Another method is to weight the distance between two proteins by the length of the shortest path between them [11]. However, the method usually generates many identical distances and leads to a "tie in proximity" problem during hierarchical clustering [6].

As a disadvantage, partitional clustering approaches and hierarchical clustering approaches can only generate separated functional modules. In fact, functional modules overlap with each other, since a protein can be included in several different functional modules to perform different functions [12,13].

Density-based clustering approaches focus on detecting highly connected subnetworks. An extreme example is to identify all fully connected subgraphs [14]. Mining fully connected subgraphs only is too strict to be used in real biological networks. A variety of alternative methods have been proposed to detect dense subgraphs by using a density threshold [15,16,17]. Recently, several density-based clustering approaches have attempted to detect overlapping functional modules [12,18]. However, such methods of detecting highly connected subnetworks neglect many peripheral proteins that connect to the core protein clusters with few links, even though these peripheral proteins may represent true interactions. In addition, biologically meaningful functional modules that do not have highly connected topologies are ignored by these approaches [6].

To mine overlapping functional modules with various topologies, we present a new graph-theoretic-based algorithm, named OMFinder. Recent results of analyzing biological networks show that highly connected proteins in the networks play an important role in evolution and likely participate in multiple biological

progresses [19,20,21,22,23]. Based on this fact, we divide the proteins into two classes of high-degree and low-degree nodes and constrain only the high-degree nodes can belong to multiple functional modules. We split the original graph G into three subgraph G_h, G_l and G_b, where G_h is a subgraph representing the relations between high-degree nodes, and G_l is a subgraph representing the relations between low-degree nodes, and G_b is a subgraph representing the relations between high-degree nodes and low-degree nodes. Each operation is only in one separated subgraph, which improves the efficiency of the algorithm effectively. We apply algorithm OMFinder to the core protein interaction network of budding yeast in DIP database. The experiment results show that our algorithm OMFinder can detect many significant functional modules effectively. Most of the identified modules overlap with each other.

2 Methods

The protein interaction network can be represented as an undirected, un-weighted graph $G(V, E)$ with proteins as a set of nodes V and interactions as a set of edges E. As the protein interaction networks are scale-free, and they are dominated only by a few nodes known as hubs. We proposed a new graph-theoretic-based algorithm for detecting overlapping functional modules in protein interaction networks. Distinguishing from other methods, we defined the graph G as a superposition of three subgraph G_h, G_l and G_b. According to the three subgraphs, we constructed a reduced graph for G. And, we constrained that only the informative nodes could belong to more than one functional modules. Based on the graph split and reduction, we developed a new algorithm, named OMFinder.

2.1 Informative Proteins Selection

Recently, the small world effect and scale-free property of protein interaction networks have been investigated extensively [19,20,21,22]. The small world is characterized by small average length of the shortest paths and large clustering coefficient. The scale-free networks follow a power law degree distribution, the probability of a node in which has a degree k is approximated by $P(k) \approx \alpha k^{-r}$ with $1 < r < 3$. The scale-free of protein interaction networks shows that only a few nodes (known as hubs) have very large degrees, while most other nodes have very few interactions. Genome-wide studies show that deletion of a hub protein is more likely to be lethal than deletion of a non-hub protein [20,21,23]. Thus, we select the nodes with large degrees as informative proteins from the protein interaction networks.

2.2 Graph Split and Reduction

The nodes in the protein interaction networks can be divided into two classes, namely informative and non-informative proteins. More precisely, we define a graph G with node set $V(G)$ that is composed of two disjoint subsets $V_h \subset V(G)$

and $V_l \subset V(G)$, where V_h is the set of high-degree nodes, and V_l is the set of low-degree nodes, and $|V_h| + |V_l| = |V(G)|$. Then the edge $e(u, v)$ in the graph G can be grouped into three classes: $e_h(u, v \in V_h)$, $e_l(u, v \in V_l)$, and $e_b(u \in V_h, v \in V_l$ or $u \in V_l$, $v \in V_h)$. Let $E_h = \{e_h\}$, $E_l = \{e_l\}$, and $E_b = \{e_b\}$. A graph G can be viewed as a superposition of three subgraph $G_h(V_h, E_h)$, $G_l(V_l, E_l)$ and $G_b(V_b, E_b)$. A simple example is illustrated in Fig. 1. Suppose we select two nodes of the highest degrees, which are marked in black in the original graph G, as informative nodes. Then the graph G can be separated into three subgraph G_h, G_l and G_b. The subgraph G_l is divided into three separated subgraphs S_1, S_2, and S_3. This is a common phenomenon known as centrality-lethality rule in protein interaction networks. If the subgraphs of G_l are reduced as nodes, named S-nodes, then the original graph G can be rebuilt as Fig. 2(a). If four nodes are selected as informative nodes from the original graph G, then graph G can be reduced as Fig. 2(b).

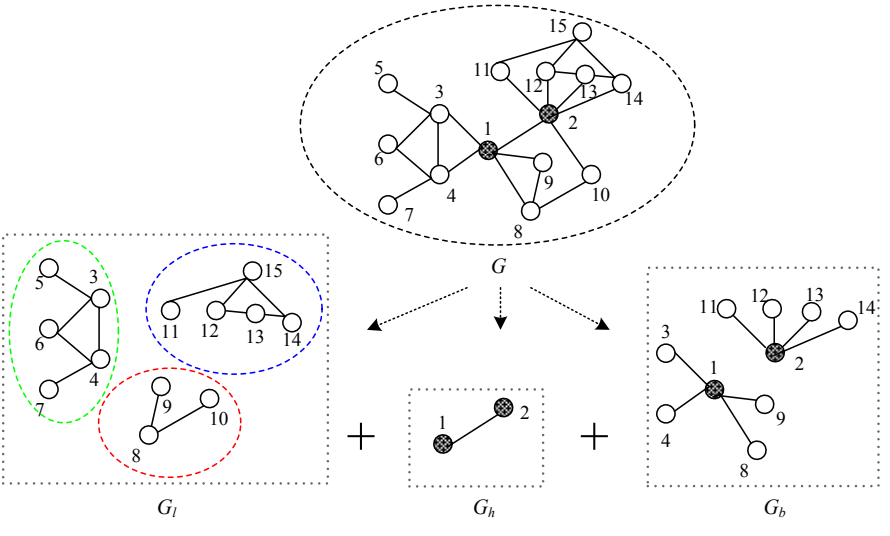


Fig. 1. An example for that a graph G is a superposition of three subgraph G_h, G_l and G_b. Node 1 and node 2 are two informative nodes of graph G, whose degrees are largest. Graph G_l is divided into three separated sub-graphs S_1, S_2, and S_3.

In Fig.2, the solid edge (e_h) connects two high-degree nodes and the dashed edge connects a high-degree node and a S-node. We construct a dashed edge between a high-degree node and a S-node, if there is one interaction between the low-degree nodes (in subgraph S) and the high-degree node in G_b. To measure how strongly the S-nodes connect to the informative nodes, we define the weight of the dashed edge as:

$$w_{hS} = \frac{|E_{hS}|}{|V_S|} \tag{1}$$

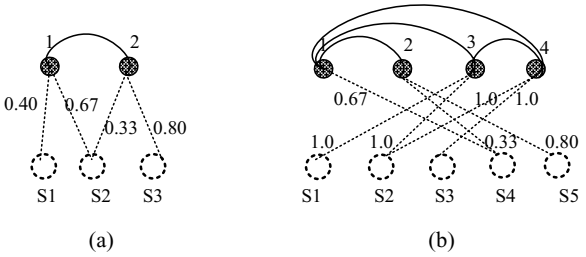


Fig. 2. The reduced graph of G. The arc edge connects two high-degree nodes, and the dashed edge connects a high-degree node and a S-node. (a) node 1 and node 2 are informative nodes, S_1={3,4,5,6,7}, S_2={8,9,10},S_3={11,12,13,14,15}; (b) node 1, node 2, node 3 and node 4 are informative nodes, S_1={5}, S_2={6},S_3={7}, S_4={8,9,10}, S_5={11,12,13,14,15}.

Here, $|E_{hS}|$ is the total number of edges between the high-degree node and the low-degree nodes in subgraph S, and $|V_S|$ is the number of nodes in subgraph S.

In biological networks, the high-degree nodes act as hubs and are essential to the networks. Jeong and colleagues analyzed the topologies and functions of 43 metabolic networks of different organisms. They found that all the 43 metabolic networks were scale-free and were dominated by the same highly connected substrates, while less connected substrates preferentially served as the educts or products of species-specific enzymatic activities [19]. Most of the substrates were only concerned with one or two metabolic reactions, only a few of substrates were concerned with multiple metabolic reactions. For protein interaction network, it is also a scale-free network and its highly connected nodes have the same property. We obtained protein lethality data from the MIPS database [24]. For the essential proteins annotated in FunCat [25], more than 80% have two or more annotations. Most of the highly connected nodes in protein interaction networks are essential. Thus, in the protein interaction networks, it is more likely for the highly connected proteins having multiple functions than the less connected proteins. In the reduced graph model, we constrained the S-nodes could only be separated into one module, and the high-degree nodes could be separated into multiple modules. Since several highly connected proteins may be concerned with the same biological progress together, we first group them by enumerating all the fully connected subgraphs in G_h. Then the predigested graphs in Fig. 2 can be reduced to the bipartite graphs showed in Fig.3.

We define the node reduced from the fully connected subgraph as C-node. The weight of the relation between a C-node and a S-node is defined as:

$$W_{CS} = \sum_{h \in C} w_{hS} \tag{2}$$

A S-node may be relate to several C-nodes. To constrain each S-node belongs to one functional module, we only construct an edge between the S-node and a C-node when the weight of the relation between them is maximum. Then, each S-node has only one C-node connecting to it. In contrast, a C-node may have

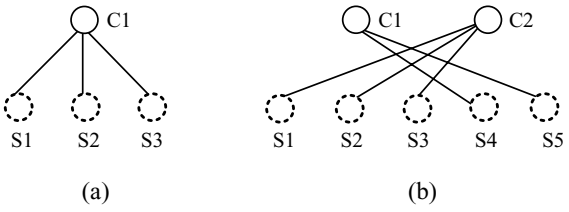


Fig. 3. The bipartite graph H reduced from Fig.2. The high-degree nodes are grouped by enumerating the fully connected subgraphs in G_h. (a) Node 1 and node 2 are informative nodes, C_1={1,2}; (b) Node 1, node 2, node 3 and node 4 are informative nodes, C_1={1,2}, C_2={1,3,4}.

several S-nodes connecting to it. Then, the separated subgraphs in the bipartite graph H are the functional modules.

3 Experiments and Results

We downloaded the budding yeast core protein interaction network (version ScereCR20070107) from DIP, the Database of Interacting Proteins [26]. We removed all the self-connecting interactions and the repeated interactions from the original network. The final core protein interaction network includes 2528 yeast proteins and 5734 interactions. We use a parameter PI (***P***ercentage of ***I***nformative proteins) to control the number of the informative nodes selected.

3.1 Identification of Overlapping Modules

We implemented OMFinder to analyze the core protein interaction network. By changing the values of parameter PI from 20% to 40%, we achieved five different output sets of modules from the protein interaction networks. As shown in Table 1, the number of identified modules with $size \geq 3$ was increasing with the increase of PI. On the contrary, the number of identified modules with $size \geq 8$ was decreasing as PI increased. The average size of the identified modules and the size of the biggest module were both decreased as PI increased. This showed the modules identified by OMFinder became more and smaller when PI increased.

Most of the identified modules shared common proteins. To evaluate their overlapping rate, we counted the number of the appearances across different modules for each protein. The average overlapping rates of identified modules with different values of PI were shown in Table 1. As PI increased, the average overlapping rate was slightly increased.

Cho, Hwang, and their colleagues showed that discarding the sparsely connected proteins could be a fatal decision which might loose the important biological information [27,28]. To evaluate how many proteins neglected by the identified modules, we define the discard rate (Dr), as shown in formula (3).

Table 1. The effect of parameter PI on clustering

Parameter	Number of the identified modules			Average	Max	Overlapping
PI	$size \geq 3$	$size \geq 5$	$size \geq 8$	size	size	rate
$PI = 20\%$	746	319	121	5.68	94	1.58
$PI = 25\%$	886	346	107	5.14	45	1.73
$PI = 30\%$	1024	345	95	4.74	39	1.84
$PI = 35\%$	1143	344	68	4.44	39	2.13
$PI = 40\%$	1263	323	55	4.22	37	2.05

$$Dr = \frac{|V| - |\cup M_i|}{|V|} \tag{3}$$

where $|V|$ was the total number of proteins in the network, $|\cup M_i|$ was the number of proteins included in all the identified modules with size larger than a given threshold. Since a module with $size = 2$ only represents one interaction with little information, a significant functional module should include at least 3 proteins. The discard rates of the identified modules generated by OMFinder using different values of parameter PI were shown in Fig.4. As shown in Fig.4, our method OMFinder had a very low discard rate, which was lower than 10% for the identified modules with size equal or larger than 3. However, CFinder and Maximal Clique both had a very high discard rate of more than 50%. If only the modules with $size \geq 5$ were considered, there were approximatively 90% proteins neglected by Maximal Clique.

3.2 Statistical Assessment and Functional Annotation

The P-value from hypergeometric distribution was often used to estimate whether a given set of proteins was accumulated by chance. It has been used as a criteria to assign each identified module a main function [7,16,22]. Here, we also calculated P-value for each identified module and assigned a function category to it when the minimum P-value occurred. The computing formula of P-value [7,16,22] was defined as:

$$P = 1 - \sum_{i=0}^{k-1} \frac{\binom{|F|}{i}\binom{|V|-|F|}{|M|-i}}{\binom{|V|}{|M|}} \tag{4}$$

where $|M|$ was the number of proteins in an identified module, $|F|$ was the number of proteins in a reference function, and k was the number of common proteins between the functional group and the identified module. Low P-value indicated that the module closely corresponded to the function because the network had a lower probability to produce the module by chance [13].

The functional classification of proteins used in this paper was collected from the MIPS Functional Catalog (FunCat) database. FunCat [25] was an annotation scheme of tree-like structure for the functional description of proteins. There

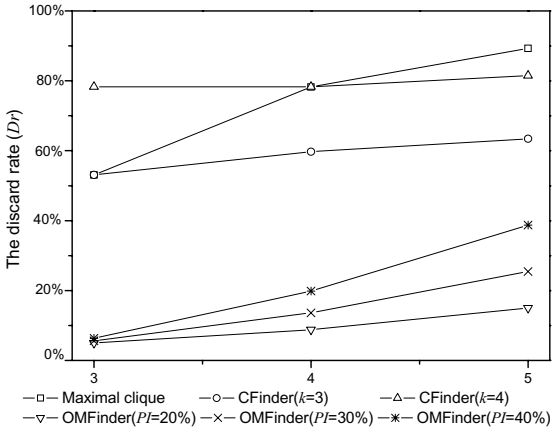


Fig. 4. The comparison of discard rates of OMFinder and other two methods: CFinder and Maximal Clique

were up to 6 levels of increasing specificity and 1360 functional categories in FunCat. We obtained 215, 219, 205, 181 and 159 modules with $size \geq 6$ when using PI=20%, 25%, 30%, 35%, and 40%. The number of the identified modules ($size \geq 6$) with $P < 0.01$ and with $P < 0.001$ generated by different values of PI was shown in Table 2.

Table 2. The number of modules ($size \geq 6$) generated by OMFinder using different values of PI with $P < 0.01$ and $P < 0.001$, respectively

Parameter	$PI = 20\%$	$PI = 25\%$	$PI = 30\%$	$PI = 35\%$	$PI = 40\%$
Number of all modules	215	219	205	181	159
Number of modules($P < 0.01$)	212	214	200	176	153
Number of modules($P < 0.001$)	189	195	178	153	113

For all the identified modules generated with different values of PI, there were more than 96.2% and 83% modules matching well with known functional categories with $P < 0.01$ and $P < 0.001$, respectively. Table 3 showed annotations for some identified modules ($size \geq 10$) with $P < 1.0 \times 10^{-10}$, where PI=25% was used.

3.3 Accuracy Analysis

Recall and precision are two important aspects to estimate the performance of algorithms for detecting functional modules. Recall is the fraction of the true-positive predictions out of all the true predictions, and precision is the fraction of the true-positive predictions out of all the positive predictions. The calculation formulae [13] of recall and precision are:

Table 3. Annotations of the identified modules ($size \geq 10$) with $P < 1.0 \times 10^{-10}$. All the identified modules are generated by using $PI = 25\%$.

ID	Size	P-value	Function	Unknown proteins
1	21	$< 1.00 \times 10^{-30}$	mitochondrial transport	YJL064W
2	20	$< 1.00 \times 10^{-30}$	rRNA processing	-
3	15	$< 1.00 \times 10^{-30}$	electron transport	YBR281C;YGR210C
4	11	1.11×10^{-16}	chromosome condensation	-
5	17	2.22×10^{-16}	rRNA synthesis	YIL141W;YJR087W
6	25	1.55×10^{-15}	general transcription activities	YLR123C;YMR102C;YHL023C
7	13	3.33×10^{-15}	microtubule cytoskeleton	-
8	25	4.44×10^{-15}	DNA repair	YJL043W; YFL042C
9	16	2.67×10^{-13}	enzymatic activity regulation /enzyme regulator	YLR190W
10	16	6.31×10^{-13}	proteasomal degradation (ubiquitin/proteasomal pathway)	-
11	13	9.06×10^{-13}	metabolism of energy reserves (e.g. glycogen, trehalose)	-
12	11	4.80×10^{-12}	cell wall	YFR044C
13	17	7.22×10^{-12}	regulation of nitrogen utilization	YIL152W;YDR078C;YLR376C; YHL006C
14	21	6.79×10^{-11}	splicing	YGR021W; YPL105C
15	15	7.28×10^{-11}	vacuole or lysosome	-
16	10	7.65×10^{-11}	perception of nutrients and nutritional adaptation	Q06966

$$recall = \frac{|M \cap F_i|}{|F_i|} \tag{5}$$

$$precision = \frac{|M \cap F_i|}{|M|} \tag{6}$$

Here, F_i is a functional category mapped to module M. The proteins in functional category F_i are considered as true predictions, the proteins in module M are considered as positive predictions, and the common proteins between F_i and M are considered as true positive predictions. It is obvious that the larger module is likely to have higher recall and lower precision. If we generate all the proteins in one module, then its recall will be equal to 1. In contrast, the module with smaller size tends to have higher precision and lower recall. As an extreme case, if we generate a single protein as one module, then we have the maximum value of precision. In general, f-measure is used as a harmonic mean of precision and recall. The f-measure [13] is defined as formula (7).

$$f - measure = \frac{2 * precision * recall}{precision + recall} \tag{7}$$

For each identified module, we calculated its f-measure to assess its accuracy. As shown in Fig.5, for the same f-measure, the number of the identified modules generated by OMFinder was all more than that generated by CFinder, the former

was about five times more than the latter. Though the number of the identified modules generated by Maximal Clique was close to those generated by OMFinder for the same f-measure, Maximal Clique discard too many proteins.

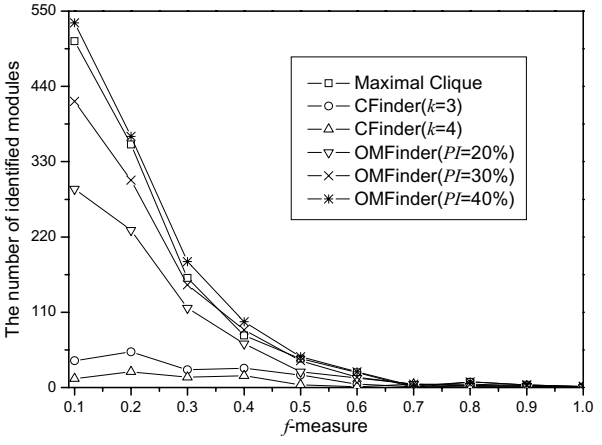


Fig. 5. The number of identified modules with respect to $f-meaure \geq 0.1, 0.2, \cdots, 1.0$

4 Conclusions

Functional modules play a special role in biological networks, which are relatively independent units performing certain biological functions. Many graph clustering methods have been developed to detecting functional modules in protein interaction networks. However, most of the previous methods can not detect the overlapping functional modules by generating separate subraphs. And, a few existed methods for identifying overlapping modules focused on detecting highly connected subgraphs, which neglected many peripheral proteins.

In this paper, we present a new graph-theoretic-based algorithm for identifying overlapping functional modules in protein interaction networks. We divide the proteins into two classes, namely high-degree and low-degree nodes, respectively. Based on the fact that highly connected proteins in biological networks play an important role in evolution and likely participate in multiple biological progresses, we constrain that only the high-degree nodes can belong to multiple functional modules. We split the original graph G into three subgraph G_h, G_l and G_b. Each operation is only in one separated subgraph, which improves the efficiency of the algorithm effectively. Our algorithm OMFinder is implemented in C++. We applied algorithm OMFinder to the core protein interaction network of budding yeast in DIP database. Many significant functional modules were detected. Of all the 219 identified modules with $size \geq 6$ ($PI = 25\%$), more than 96.2% corresponded to $P < 0.01$, and more than 86.8% corresponded to $P < 0.001$. We predicted functions for previous unknown proteins by assigning the identified modules a main function with the lowest P-value. We identified

more overlapping functional modules with high recall and precision than previous methods CFinder. Most importantly, our algorithm OMFinder can cover most of the proteins in the network, which neglect few peripheral proteins. As a new graph-theoretic method, we think that it will be helpful to detect functional modules and to analyze the topologies of biological networks.

Acknowledgments. The authors wish to thank Adamcsek B., Palla G., Farkas I., Derenyi I., and Vicsek T for sharing their program of CFinder.

References

1. Asur, S., Ucar, D., Parthasarathy, S.: An ensemble framework for clustering protein-protein interaction networks. ISMB/ECCB 23, 29–40 (2007)
2. Hartwell, L.H., et al.: From molecular to modular cell biology. Nature 402, 47–52 (1999)
3. Barabasi, A.L., Oltvai, Z.N.: Network biology: understanding the cell's functional organization. Nat. Res. 5, 101–114 (2004)
4. Chen, J.C., Yuan, B.: Detecting functional modules in the yeast protein-protein interaction network. Bioinformatics 22, 2283–2290 (2006)
5. Rives, A.W., Galitski, T.: Modular organization of cellular networks. Proc. Natl. Acad. Sci. 100, 1128–1133 (2003)
6. Luo, F., et al.: Modular organization of protein interaction networks. Bioinformatics 23, 207–214 (2007)
7. King, A.D., Pržulj, N., Jurisica, I.: Protein complex prediction via cost-based clustering. Bioinformatics 20, 3013–3020 (2004)
8. Hartuv, E., Shamir, R.: A clustering algorithm based graph connectivity. Information Processing Letters, 175–181 (2000)
9. Girvan, M., Newman, M.E.: Community structure in social and biological networks. Proc. Natl. Acad. Sci. 99, 7821–7826 (2002)
10. Pereira-Leal, J.B., et al.: Detection of functional modules from protein interaction networks. Proteins: Struct. Func. Bioinformatics 54, 49–57 (2004)
11. Arnau, V., et al.: Iterative cluster analysis of protein interaction data. Bioinformatics 21, 364–378 (2005)
12. Adamcsek, B., Palla, G., Farkas, I., Derenyi, I., Vicsek, T.: CFinder: locating cliques and overlapping modules in biological networks. Bioinformatics 22, 1021–1023 (2006)
13. Cho, Y.R., Hwang, W., Ramanathan, M., Zhang, A.: Semantic integration to identify overlapping functional modules in protein interaction networks. BMC Bioinformatics, 8–265 (2007)
14. Spirin, V., Mirny, L.A.: Protein complexes and functional modules in molecular networks. Proc. Natl. Acad. Sci., 12123–12128 (2003)
15. Bader, G.D., Hogue, C.W.: An automated method for finding molecular complexes in large protein interaction networks. BMC Bioinformatics 4(1) (2003)
16. Altaf-Ul-Amin, M., et al.: Development and implementation of an algorithm for detection of protein complexes in large interaction networks. BMC Bioinformatics, 7(207) (2006)
17. Pei, P., Zhang, A.: A seed-refine algorithm for detecting protein complexes from protein interaction data. IEEE Transactions on Nanobioscience 6, 43–50 (2007)

18. Palla, G., et al.: Uncovering the overlapping community structure of complex networks in nature and society. Nature 435, 814–818 (2005)
19. Jeong, H., Tombor, B., Albert, R., Oltvai, Z.N., Barabasi, A.L.: The large-scale organization of metabolic networks. Nature 407, 651–654 (2000)
20. Jeong, H., Mason, S., Barabási, A., Oltvai, Z.: Lethality and centrality in protein networks. Nature 411, 41–42 (2001)
21. Yook, S., Oltvai, Z., Barabasi, A.: Functional and topological characterization of protein interaction networks. Proteomics 4, 928–942 (2004)
22. Pržulj, N., Wigle, D.A., Jurisica,I.: Functional topology in a network of protein interactions. Bioinformatics 20(3), 340–348 (2004)
23. Ucar, D., Asur, S., Catalyurek, U., Parthasarathy, S.: Improving functional modularity in potein-protein interactions graphs using Hub-induced subgraphs. In: Fürnkranz, J., Scheffer, T., Spiliopoulou, M. (eds.) PKDD 2006. LNCS (LNAI), vol. 4213, Springer, Heidelberg (2006)
24. Mewes, H.W., et al.: MIPS: analysis and annotation of proteins from whole genome in 2005. Nucleic Acid Research 34, 169–172 (2006)
25. Ruepp, A., et al.: The FunCat, a functional annotation scheme for systematic classification of proteins from whole genomes. Nucleic Acid Research 32, 5539–5545 (2004)
26. `http://dip.doe-mbi.ucla.edu/`
27. Hwang, W., Cho, Y.R., Zhang, A., Ramanathan, M.: A novel functional module detection algorithm for protein- protein interaction networks. Algorithms for Molecular Biology 12, 1–24 (2006)
28. Cho, Y.R., Hwang, W., Zhang, A.: Identification of overlapping functional modules in protein interaction networks: information flow-based approach. In: Perner, P. (ed.) ICDM 2006. LNCS (LNAI), vol. 4065, Springer, Heidelberg (2006)

Identification of Transcription Factor Binding Sites in Promoter Regions by Modularity Analysis of the Motif Co-occurrence Graph

Alexandre P. Francisco, Arlindo L. Oliveira, and Ana T. Freitas

INESC-ID/IST, Technical University of Lisbon, Portugal
{aplf,aml,atf}@inesc-id.pt

Abstract. Many algorithms have been proposed to date for the problem of finding biologically significant motifs in promoter regions. They can be classified into two large families: combinatorial methods and probabilistic methods. Probabilistic methods have been used more extensively, since their output is easier to interpret. Combinatorial methods have the potential to identify hard to detect motifs, but their output is much harder to interpret, since it may consist of hundreds or thousands of motifs. In this work, we propose a method that processes the output of combinatorial motif finders in order to find groups of motifs that represent variations of the same motif, thus reducing the output to a manageable size. This processing is done by building a graph that represents the co-occurrences of motifs, and finding communities in this graph. We show that this innovative approach leads to a method that is as easy to use as a probabilistic motif finder, and as sensitive to low quorum motifs as a combinatorial motif finder. The method was integrated with two combinatorial motif finders, and made available on the Web.

1 Introduction

An important open problem in computational biology is related with the accurate identification of biologically meaningful nucleotide sequences in promoter regions, that correspond to loci of attachment of transcription factors. These well conserved regions are usually referred to as consensus sequences or motifs. Motif finding is the problem of discovering these motifs without prior knowledge of their characteristics. Motif finding has been the subject of intense research and literally hundreds of papers have been published on this topic [1].

Currently available methods for motif finding can roughly be classified in two main classes: probabilistic and combinatorial.

Probabilistic methods have been extensively used, and they identify very well the strong signals present in the data, i.e., motifs that occur in a large fraction of the sequences. They have difficulties identifying weaker signals, that correspond to motifs that are present only in a subset of sequences, possibly superimposed with stronger signals.

Combinatorial methods, on the other hand, when executed with the right parameters, can identify both strong and weak signals. They suffer, however,

I. Măndoiu, R. Sunderraman, and A. Zelikovsky (Eds.): ISBRA 2008, LNBI 4983, pp. 220–231, 2008.

from a significant drawback. When executed with parameters that allow them to identify motifs that are present in only a small fraction of the sequences, they will deluge the user with a large, possibly huge, number of motifs, that correspond to many variations of the motifs of interest. In fact, since motifs are not perfectly conserved, many variations of the most common motifs will be reported by a combinatorial motif finder, since these variations will appear in a significant fraction of the sequences.

In this work, we propose a method for the identification of motifs that combines the advantages of probabilistic motif finders (easy to use, no parameters required) and combinatorial motif finders (ability to identify even the weaker signals) while avoiding the disadvantages of both.

We achieve this by post-processing the results of combinatorial motif finders, and identifying the motifs that are variations of the same signal. These motifs are then combined into a composed representation, and a position weight matrix (PWM) is generated for that set of motifs. The identification of the motifs that are, in reality, variations of the same motif, is done by computing the modules (or communities) in a graph. This graph has one node for each motif found, and one edge between two motifs if they have significant occurrence overlap.

2 Basic Concepts and Related Work

2.1 Motif Finders

The most used probabilistic algorithms for motif finding are based on the application of the Expectation-Maximization [2] method (EM) like PROJECTION [3] and MEME [4] or its stochastic counterpart, Gibbs sampling [5]used by AlignACE [6], BioProspector [7] and GibbsDNA [5]. These methods use a two-phase iterative procedure where, in the first step the likeliest occurrences of the motif are identified, based on a model computed in the previous iteration. The second step adjusts the model for the motif (usually a position weight matrix) based on the occurrences determined in the previous step. In the first iteration the parameters of the initial model are usually set randomly. This iterative procedure converges usually in a rapid way to motifs that are present, possibly with mutations, in a large fraction of input sequences. They report their results in the form of PWMs, that represent directly the obtained model.

Combinatorial methods, which extract motifs consisting of plain nucleotide sequences work by enumerating the possible patterns, either explicitly or implicitly [8,9],and counting their quorum. Consider a set of sequences under analysis, $\mathcal{S} = \{S_1, S_2, \ldots, S_t\}$. The objective is to find motifs within a range of lengths $l_{\min}, \ldots, l_{\max}$, which occur on $q \leq t$ of the sequences in S with at most e mismatches, i.e., having at most e nucleotide substitutions. Algorithms that look for complex motifs have also been proposed [10,11]. Complex motifs are built of two or more simple motifs, spaced by a number of bases that falls within a specific range.

For this work we selected the combinatorial motif finders MUSA [12] and RISO [10]. MUSA is an algorithm that does not require the user to specify

parameters (such as box lengths and distances between boxes) in order to extract motifs. This method relies on a biclustering algorithm that operates on a matrix of co-occurrences of small motifs. Requiring as input a list of gene promoter sequences, MUSA returns the list of structured or simple motifs found, ordered by their p-value, and their quorum. RISO is a complex motif extraction tool. It searches for complex motifs with certain characteristics specified by the user, through the assignment of a set of parameters such as the number and sizes of the boxes that form the structured motif, the distances between them and the minimum quorum expected. RISO returns the list of motifs found and their corresponding quorums.

2.2 Motif Clustering

The idea of finding groups, or clusters, of motifs, in order to simplify the binding site studies and reduce the redundancy of the patterns found by motif finders is not new, and, indeed, has been proposed independently. Examples of tools that perform this clustering are MatAlign [13] and Stamp [14].

Although there are differences in the implementation, these and other existing methods work by defining a distance between two motifs and applying standard clustering methods to find motifs with similar patterns. The distance is typically obtained using dynamic programming to compute the best alignment between two motifs.

While this approach works well in some cases, it has some strong limitations. In particular, this approach is not able to identify that two motifs are part of the same pattern if they are poorly aligned, even if they represent different parts of the same, larger, motif. For instance, motifs ACCGTG and TGATTT may be frequent because the larger motif ACCGTGATTT is frequent, but no significant alignment will be found between motifs ACCGTG and TGATTT, specially if, for some reason, the larger motif is not identified.

The approach we propose avoids this difficulty by ignoring the actual pattern of the motifs, and considering only the sequences and positions where they occur. A significant amount of co-occurrence means that two motifs are linked, even though they may not be very similar.

For this, we need a method that finds communities in graphs of motifs, i.e., groups of motifs that are tightly linked by many co-occurrences.

2.3 Finding Communities in Graphs

Many algorithms have been developed to tackle the problem of finding communities in graphs [15,16,17,18].Probably the best-known is the one proposed by Girvan and Newman [15] based on the betweenness centrality measure which runs in $O(|E|^2|V|)$ time, or $O(|V|^3)$ time for sparse graphs. However, it is no longer the most efficient and effective clustering algorithm. More recently Newman [17] has proposed a faster algorithm based on the greedy optimization of the modularity [16] which is substantially faster. It runs in $O((|E| + |V|)|V|)$, or $O(|V|^2)$ on sparse graphs. However, the running time of this algorithm can be improved by

exploiting some properties of the optimization problem and using more sophisticated data structures. Thus, Clauset *et al.* [18] proposed a greedy algorithm which runs in $O(|E|d \log |V|)$, where d is the depth of the "dendrogram" which describes the community structure. On sparse graphs with a hierarchical community structure this algorithm runs on average in $O(|V| \log^2 |V|)$ time. In what follows, we name this algorithm as *CNM (Clauset-Newman-Moore) algorithm.*

The concept of modularity is central to this problem [16]. Modularity is a property of the graph and of a specific division of the graph into communities. It measures the quality of the division by evaluating the number of edges within communities and the number of edges that connect vertices in different communities. Suppose the vertices are divided into k communities and let $1 \leq c_m \leq k$ denote the community where vertex $m \in V$ belongs. The *adjacency matrix* A of G and the *degree* d_m of a vertex $m \in V$ are respectively defined as

$$A_{mn} = \begin{cases} 1 \text{ if } (m,n) \in E, \\ 0 \text{ otherwise} \end{cases} \quad \text{and} \quad d_m = \sum_{n \in V} A_{mn}. \tag{1}$$

We define the *modularity* Q of G with respect to the given division as

$$Q = \frac{1}{2|E|} \sum_{m,n \in V} \left[A_{mn} - \frac{d_m d_n}{2|E|} \right] \delta(c_m, c_n), \tag{2}$$

where the δ-function is such that $\delta(i,j) = 1$ if $i = j$ and $\delta(i,j) = 0$ otherwise. We note that the above sum runs over all possible pairs of vertices. Therefore, each edge is summed twice. If we split the sum in two terms,

$$\frac{1}{2|E|} \sum_{m,n \in V} A_{mn} \delta(c_m, c_n) \quad \text{and} \quad \frac{1}{2|E|} \sum_{m,n \in V} \frac{d_m d_n}{2|E|} \delta(c_m, c_n), \tag{3}$$

then the first term is the fraction of edges that fall within the communities, and the second term is the expected fraction of edges within the communities if the edges were randomly distributed respecting the vertices degrees. In particular, if the edges were randomly placed as mentioned, $d_m d_n / |E|$ is the probability of the existence of an edge between vertices m and n.

Thus, modularity measures the fraction of edges that connect vertices in the same component minus the expected value of the same quantity in a graph with the same components but random connections between the vertices [16]. Values near 1, the maximum value of Q, indicate strong community structure. Typically, values for graphs underlying common networks with known community structure are in the range from 0.3 to 0.7.

The CNM algorithm operates by finding the changes in Q which result from merging each pair of communities. It chooses the largest of such possible changes in a greedy way and performs the merging. Let ΔQ_{ij} be the change in Q that results from merging the communities i and j. Initially each vertex $m \in V$ is a community, by equation 2,

$$Q = -\frac{1}{2|E|} \sum_{m \in V} \frac{d_m d_m}{2|E|}. \tag{4}$$

And, for each community i and for each pair of communities i, j, we set

$$A_i = \frac{d_m}{2|E|}, \quad \text{and} \quad \Delta Q_{ij} = \begin{cases} \frac{1}{|E|} - 2A_iA_j & \text{if } i,j \text{ are connected}, \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

where m is the unique vertex in the community i. Thus, the main loop of the CNM algorithm consists in

1. select the largest ΔQ_{ij} and increment Q accordingly,
2. join the corresponding communities and, assuming that community i is merged into community j, update the ΔQ and the A values as follows:

$$\Delta Q_{jk} = \begin{cases} \Delta Q_{ik} + \Delta Q_{jk} & \text{if } k \text{ is connected to } i \text{ and } j, \\ \Delta Q_{ik} - 2A_jA_k & \text{if } k \text{ is connected to } i \text{ but not to } j, \\ \Delta Q_{jk} - 2A_iA_k & \text{if } k \text{ is connected to } j \text{ but not to } i; \end{cases} \tag{6}$$

$$A_j = A_j + A_i. \tag{7}$$

3 Finding Co-occurring Motifs

As described, our method builds a motif relation graph and finds communities of motifs, i.e., subgraphs such that the density of edges within it is greater than the density of edges between its vertices and those outside it. Each community is then processed in order to obtain the associated Position Weight Matrix (PWM).

3.1 Building the Relation Graph

Let $\mathcal{S}$ be the set of sequences and $\mathcal{M}$ be the set of motifs found in $\mathcal{S}$. For each $m \in \mathcal{M}$ and $s \in \mathcal{S}$, let $l(m, s)$ be the list of positions in s where m occurs. We say that $m, n \in \mathcal{M}$ *overlap* in a sequence $s \in \mathcal{S}$ if $x \in l(m, s)$ and $y \in l(n, s)$ exist such that one of the following two conditions verifies:

$$x < y < x + |m|; \qquad y \leq x < y + |n|. \tag{8}$$

In such case we say that motifs m and n overlap in s with a *shift* σ equal to $y - x$.

In our method, we assume that a minimum overlap $0 < \alpha_o \leq 1$ and a minimum quorum $0 < \alpha_q \leq 1$ are given as parameters. The quorum represents the fraction of the number of common sequences in which a given pair of overlapping motifs must occur to be considered. Therefore, given two motifs $m, n \in \mathcal{M}$ we define the *minimum shift* and the *maximum shift*, for m and n, respectively, as

$$\sigma_{\min} = \alpha_o \min\{|m|, |n|\} - |n| \quad \text{and} \quad \sigma_{\max} = |m| - \alpha_o \min\{|m|, |n|\}. \tag{9}$$

To ensure that m and n overlap in a given sequence $s \in \mathcal{S}$ with at least $\alpha_o \min\{|m|, |n|\}$ common characters, we must check that such overlap occurs with a shift σ such that $\sigma_{\min} \leq \sigma \leq \sigma_{\max}$.

The *relation graph* G is a tuple $\langle V, E \rangle$ where V and E are defined as follows. The set of vertices V is the set of motifs found, i.e., $V = \mathcal{M}$. The set of edges

E contains every pair $(m, n) \in \mathcal{M} \times \mathcal{M}$ for which exists $\sigma \in \mathbb{N}$ such that $\sigma_{\min} \leq \sigma \leq \sigma_{\max}$ and

$$\alpha_{\mathrm{q}}|\mathcal{S}'| \leq |\{s \in \mathcal{S}' \ : \ m, n \text{ overlap in } s \text{ with shift } \sigma\}|, \tag{10}$$

where $\mathcal{S}' \subseteq \mathcal{S}$ is the set of sequences in which both m and n occur.

3.2 Implementation of the CNM Algorithm

In its original version, the CNM algorithm iterates until a negative ΔQ_{ij} is selected, and stops when all vertices belong to the same community. However, in our case, the relation graph G may not be connected, and the selection of a negative value is an admissible stop condition because after that Q can only decrease.

The bound $O(|E|d \log |V|)$ in the running time can only be achieved if advanced data structures are used [18]. In our implementation, we store the ΔQ_{ij} values in a red-black tree for each community. Additionally, we maintain these values in binary heaps. Therefore the insertion, the selection and the maximum extraction can be done in $O(\log |V|)$ [19]. We also use the well known union-find data structure [19] to track the vertices in each community. Because the relation graph G is, in general, sparse, the CNM algorithm complexity is almost linear in the number of motifs.

3.3 Computing and Ranking the PWM of a Community of Motifs

By applying the CNM algorithm to the motif relation graph and choosing the partition which grants maximum modularity, we obtain a set of communities of motifs. The third and final step of our method consists in processing each of these communities and computing the PWM for each one.

Thus, let $\mathcal{C} \subseteq \mathcal{M}$ be a community of motifs found in the graph G. First, we align the motifs in $\mathcal{C}$, which is simple because we already know the relative shift from the graph construction, and we compute the length of the PWM for this community. Second, for each edge $(n, m) \in E$ and using the best shift for equation 10, i.e., the shift which maximizes the right side of equation 10, we update the corresponding columns of the PWM by checking the symbols in the sequences where the pair of motifs occur, i.e, the sequences in the set S' in equation 10.

Each community gets assigned a p-value that correspond to the lowest p-value of the motifs in that community. This p-value is used to rank the communities and corresponding PWMs. For each community a quorum is also computed. This quorum corresponds to the average number of sequences that support each edge in the community structure.

The method developed was implemented in `C`, including the CNM algorithm and all data structures. The resultant tool was integrated with two motif finders, MUSA and RISO , and made available through the DISCOVERER platform in the YEASTRACT database [20]. Given that the complexity of the CNM algorithm is almost linear for sparse graphs, the computation of the relation graph is the most computational demanding step of our method, taking $\Omega(|M|^2)$

time. For all test examples, which have at most 3000 statistically significant motifs, we were able to compute motif communities and corresponding PWMs in less than one minute in a common workstation.

4 Results

In this section we only present the results obtained, with the motif finder MUSA, for the first two datasets described in Table 1. More detailed results for both motif finders and for all datasets are available, as supplementary material, at http://kdbio.inesc-id.pt/mat/isbra08.

To test the ability of the proposed method to find relevant motif communities, four real biological datasets were used. These datasets correspond to different sets of promoter sequences of *Sacharomyces cerevisiae* genes.

For all datasets the MUSA algorithm was executed with the default parameter values: $\lambda = 4$, $\epsilon = 1$ and the quorum equal to 30%. The motifs reported were ranked in accordance with their statistical significance. Motifs that have a p-value smaller than 10^{-3} were considered statistically significant and selected for further processing. To build the relation graph, for these motifs, the default values of α_{o}, the minimum overlap, and α_{q}, the minimum quorum, were 0.4 and 0.7, respectively.

Table 1 summarizes the results obtained. For each dataset it shows the number of sequences (N. seq), that were used to search for over-represented motifs, the total number of motifs found by the motif finder (T. mot), the number of motifs considered for processing (N. mot), the total number of edges in the relation graph (N. edg), the number of communities identified (N. com) and the modularity value achieved (Modul). It is clear from this table that the method

Table 1. Datasets content and results statistics

Datasets	*N.seq*	*T.mot*	*N.mot*	*N.edg*	*N.com*	*Modul*
DeRisi [21]	25	1647	204	299	89	0.80
Aft2p [22]	193	2176	2026	3397	202	0.84
Yap1p [23]	225	2150	2065	3541	168	0.84
2,4D [24]	486	2088	2045	3143	271	0.86

effectively reduces the number of motifs that need to be analyzed, by up to one order of magnitude.

The first dataset, identified as the DeRisi set, corresponds to a list of 25 genes that were up-regulated in response to the expression of a point mutation in the PDR1 gene, that encodes a transcription factor (TF) involved in Pleiotropic Drug Resistance in yeast [21]. Due to the experimental procedure used, this set correspond to a small and very well characterized set of genes where the Pdr1p binding site can be easily identified.

For this set, using the default input parameters and considering both strands, the MUSA algorithm identified 1647 over-represented motifs. From these, only

Table 2. The top 15 motifs reported by MUSA for the DeRisi dataset

ID	*Motif*	*Quorum*	$P - value$
1	TCCGTGGA	12 of 25	2.79106e-17
2	TCCACGGA	12 of 25	2.79106e-17
3	AAGA (17,19) TTTC	18 of 25	3.57327e-16
4	GAAA (17,19) TCTT	18 of 25	3.57327e-16
5	CCGT (1,3) GAAA	13 of 25	7.22785e-15
6	TTTC (1,3) ACGG	13 of 25	7.22785e-15
7	CCACGGA	14 of 25	8.57847e-15
8	TCCGTGG	14 of 25	8.57847e-15
9	AAAA (4,6) AAAT	25 of 25	1.31737e-14
10	ATTT (4,6) TTTT	25 of 25	1.31737e-14
11	CCACGGAA	11 of 25	1.45924e-14
12	TTCCGTGG	11 of 25	1.45924e-14
13	AACA (43,45) CCTC	11 of 25	2.40999e-12
14	GAGG (43,45) TGTT	11 of 25	2.40999e-12
15	CAAAAG (3,5) AAAT	9 of 25	4.18278e-12

204 motifs were classified as statistically significant and considered for further processing. Table 2 presents the first 15 motifs reported by MUSA. In this list, motifs 1 and 7 correspond to instances of the TF binding site of interest. Although these motifs are well positioned in the motifs list, it is possible to verify that they are only present in at most half of the input sequences. This low quorum hides the real importance of this binding site, something that is not expected in this particular dataset where all the genes were up-regulated by Pdr1p.

By inspecting Table 2 it is also possible to note that there is a large number of complex motifs in the top 15 motifs reported. For space reasons, we will not describe, in this article, the way complex motifs are handled, although a trivial extension of the method exists and has been implemented.

Figure 1 shows the 14 motifs which contribute to the most significant community obtained. The PWM description of this community is also presented. By inspecting this figure it is possible to see that the quorum of this community is 97%. This value is very important since it reflects the real importance of this binding site in this set. The community quorum is also an important feature in the evaluation of the community importance. Figure 2 shows the PWMs logos for the two most significant motif communities identified.

If a search for documented TFs binding sites is performed in YEASTRACT database using the PWM of the first community, the best match will be one of the documented Pdr1p-binding sites. If this search is also performed for the second most important community, shown in Figure 2, again one of the documented Pdr1p-binding sites will be found. This second motif is similar to the first one but not included in the same community because of the difference in the central nucleotides: an GT switched for a AC. In fact, since MUSA searched for motifs in both strands, this second community roughly contains the reverse complement of the motifs contained in the first community. This community could be trivially

```
Community 1 (14 motifs) (quorum stat: min 0.71 avg 0.97 max 1.00)
    ---TCCGTGGA-      0    14    8   2.79106e-17
    ---TCCGTGG--      1    17    9   8.57847e-15
    --TTCCGTGG--      2    12    9   1.45924e-14
    ----CCGTGG--      3    26   10   2.03787e-09
    ---TCCGTG---      4    18    9   2.16867e-07
    --TTCCGT----      5    28   13   4.31347e-07
    -----CGTGG--      6    45   16   1.13802e-05
    -----CGTGGA-      7    23   10   5.08829e-05
    ----CCGTG---      8    33   13   0.000103709
    -GTTCCG-----      9    15    8   0.000258935
    ---TCCGTC---      10   16    5   0.00026753
    CCCTC-------      11   36   14   0.000360318
    ------GTGGA-      12   48   17   0.000506259
    ------GTGGAA      13   21    8   0.000797191

A: 0.26 0.11 0.12 0.05 0.01 0.00 0.00 0.00 0.00 0.00 0.90 0.63
C: 0.12 0.28 0.10 0.04 0.97 1.00 0.00 0.00 0.01 0.00 0.03 0.03
G: 0.18 0.36 0.07 0.03 0.01 0.00 1.00 0.00 0.99 0.99 0.06 0.17
T: 0.44 0.25 0.71 0.88 0.01 0.00 0.00 1.00 0.00 0.01 0.01 0.17
```

Fig. 1. Depiction of the most significant community found for the DeRisi dataset: motifs alignment and PWM description

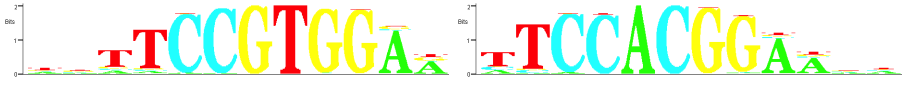


Fig. 2. Depiction of the PWM logos for the first and second most important communities identified

merged with the first one, however when reporting only one of the communities we have found out that some users do not recognize the motif they were looking for. Thus, we decided to report both.

The second dataset, identified as Aft2p, includes 193 promoter sequences of Aft2p-target genes. This TF is involved in the regulation of iron homeostasis and associated oxidative stress [25]. There is evidence supporting the direct binding of Aft2p to the promoter region of many of these 193 target-genes [26]. However, for some of them, evidence of a regulatory association is indirect, coming from the comparison of gene transcript levels in the wild type and in a mutant devoid of AFT2 [22]. In this test case the list of genes considered includes direct and indirect targets of Aft2p and, due to this fact, this TF binding site is not expected to be a strong signal in the sequences.

From a total of 2176 motifs extracted by the motif finder, only 2026 were classified as statistically significant and considered for further processing. For this set, 202 communities were identified. Table 3 presents the top 15 motifs extracted by MUSA.

Table 3. The top 15 motifs reported by MUSA for the Aft2p dataset

ID	*Motif*	*Quorum*	$P-value$
1	TTTT (13,15)CACC	64 of 193	6.54141e-34
2	ACATAT	139 of 193	1.63126e-31
3	ATATGT	139 of 193	1.63126e-31
4	CACCC	165 of 193	6.43219e-31
5	GGGTG	165 of 193	6.43219e-31
6	GAAGAA	149 of 193	6.3155e-30
7	TTCTTC	149 of 193	6.3155e-30
8	GTATAT	124 of 193	1.36158e-29
9	CATATA	130 of 193	3.35912e-29
10	AAGAAG	145 of 193	1.04215e-28
11	TATTCT	145 of 193	1.31282e-28
12	CAAGAA	148 of 193	5.27289e-28
13	TTCTTG	148 of 193	5.27289e-28
14	GCACC	155 of 193	4.69138e-27
15	GGTGC	155 of 193	4.69138e-27

Fig. 3. Depiction of the PWM logo for the Aft2p-target genes binding site

Figure 3 shows the PWM logo of the fourth most significant community found, that corresponds precisely to the documented Aft2p-binding site [22].The binding site previously described for this TF allows some variability in the peripheral nucleotides. It is interesting to note that the identified community also exhibits a central conserved core region, the motif CACCC, flanked by less conserved peripheral nucleotides. This core motif, that correspond to motif 4 in Table 3, is statistically significant and is present in 85% of the input sequences. The correspondent community still presents a better quorum, 91%.

The first three most significant communities found for this dataset were compared with the documented TF binding sites described in the YEASTRACT database. The third most significant community was also associated with the documented Aft2p-binding site. The second most significant community matched the *TATA-box*, a well characterized core promoter element also expected to be a strong signal in this dataset.

For the first community found, the alignments obtained were very poor, suggesting that there is no documented TF binding site with such characteristics. To further investigate the existence of a similar TF binding site, the original PWM was trimmed. In this case, the best match was with the binding site of the Rap1p transcription factor. This TF is described as a DNA-binding protein involved in either activation or repression of transcription, depending on binding site context. However, the trimmed PWM aligned only with a short part of the

Rap1p TF binding site.Although presumably not related with the documented Aft2p binding site or other documented TF binding site, this motif can have an important biological meaning.

5 Discussion

In this paper we proposed a methodology that assembles a list of individual simple motifs into communities of motifs, leading to a simplified analysis of the motif finders results.

For the test-cases presented, the results show that this method is able to identify the most important motif communities. In fact this approach is very useful in reducing the number of motifs to be inspected, leading to a more tractable output, easier to interpret by humans. The PWM representation of the community highlights the motifs degeneracy, being more informative than the consensus representation usually reported by combinatorial motif finders. The quorum of the community reveals the real importance of the motifs in the dataset.

Compared to the first test-case, the results obtained for the second test-case seem less precise. However the results achieved are still remarkably important. The post-processing of the motif finder results allowed the identification of the Aft2p binding site and suggested new putative binding sites. The third and fourth test-cases show that even for more noisy datasets this approach can provide interesting clues on how transcription factors interact with their target genes.

References

1. Sandve, G., Drablos, F.: A survey of motif discovery methods in an integrated framework. Biology Direct. 1(1), 11 (2006)
2. Segal, E., Sharan, R.: A discriminative model for identifying spatial cis-regulatory modules. Journal of Computational Biology 12(6), 822–834 (2005)
3. Buhler, J., Tompa, M.: Finding motifs using random projections. Journal of Computational Biology 9(2), 225–242 (2002)
4. Bailey, T., Elkan, C.: Fitting a mixture model by expectation maximization to discover motifs in biopolymers. In: Proceedings of the Second International Conference on Intelligent Systems for Molecular Biology, pp. 28–36 (1994)
5. Lawrence, C.E., Altschul, S.F., Boguski, M.S., Liu, J.S., Neuwald, A.F., Wootton, J.C.: Detecting subtle sequence signals: A Gibbs sampling strategy for multiple alignment. Science 262(5131), 208–214 (1993)
6. Roth, F.P., Hughes, J.D., Estep, P.W., Church, G.M.: Finding DNA regulatory motifs within unaligned noncoding sequences clustered by whole-genome mRNA quantitation. Nature Biotechnology 16, 939–945 (1998)
7. Liu, X., Brutlag, D.L., Liu, J.S.: BioProspector: discovering conserved DNA motifs in upstream regulatory regions of co-expressed genes. In: Pacific Symposium on Biocomputing, vol. 6, pp. 127–138 (2001)
8. Sagot, M.F.: Spelling approximate repeated or common motifs using a suffix tree. Latin 98, 111–127 (1998)
9. Pevzner, P.A., Sze, S.H.: Combinatorial approaches to finding subtle signals in DNA sequences. In: Proceedings of the International Conference on Intelligent Systems for Molecular Biology, vol. 8, pp. 269–278 (2000)

10. Carvalho, A.M., Freitas, A.T., Oliveira, A.L., Sagot, M.-F.: An efficient algorithm for the identification of structured motifs in DNA promoter sequences. IEEE Transactions on Computational Biology and Bioinformatics 3(2), 126–140 (2006)
11. Marsan, L., Sagot, M.F.: Algorithms for extracting structured motifs using a suxffix tree with an application to promoter and regulatory site consensus identification. Journal of Computational Biology 7(3-4), 345–362 (2000)
12. Mendes, N., Casimiro, A., Santos, P., Sá-Correia, I., Oliveira, A., Freitas, A.: MUSA: A parameter free algorithm for the identification of biologically significant motifs. Bioinformatics 22, 2996–3002 (2006)
13. Kankainen, M., Loytynoja, A.: MATLIGN: a motif clustering, comparison and matching tool. BMC Bioinformatics 8(1), 189 (2007)
14. Mahony, S., Benos, P.V.: STAMP: a web tool for exploring DNA-binding motif similarities. Nucleic Acids Research (2007)
15. Girvan, M., Newman, M.E.J.: Community structure in social and biological networks. Proceedings of the National Academy of Sciences 99, 7821 (2002)
16. Newman, M.E.J., Girvan, M.: Finding and evaluating community structure in networks. Physical Review E 69, 026113 (2004)
17. Newman, M.E.J.: Fast algorithm for detecting community structure in networks. Physical Review E 69, 066133 (2004)
18. Clauset, A., Newman, M.E.J., Moore, C.: Finding community structure in very large networks. Physical Review E 70, 066111 (2004)
19. Cormen, T.H., Leiserson, C.E., Rivest, R.L., Stein, C.: Introduction to Algorithms. MIT Press, Cambridge (2001)
20. Teixeira, M.C., Monteiro, P., Jain, P., Tenreiro, S., Fernandes, A.R., Mira, N.P., Alenquer, M., Freitas, A.T., Oliveira, A.L., Sá-Correia, I.: The YEASTRACT database: a tool for the analysis of transcription regulatory associations in saccharomyces cerevisiae. Nucleic Acids Research 34, D446–D451 (2006)
21. DeRisi, J., van den Hazel, B., Marc, P., Balzi, E., Brown, P., Jack, C., Goffeau, A.: Genome microarray analysis of transcriptional activation in multidrug resistance yeast mutants. FEBS Letters 470, 156–160 (2000)
22. Courel, M., Lallet, S., Camadro, J.M., Blaiseau, P.L.: Direct activation of genes involved in intracellular iron use by the yeast iron-responsive transcription factor Aft2 without its paralog Aft1. Molecular Cell Biology 25(15), 6760–6771 (2005)
23. Cohen, B.A., Pilpel, Y., Mitra, R.D., Church, G.M.: Discrimination between paralogs using microarray analysis: application to the Yap1p and Yap2p transcriptional networks. Molecular Biology of the Cell 13(7), 1608–1614 (2002)
24. Teixeira, M.C., Fernandes, A.R., Mira, N.P., Becker, J.D., Sá-Correia, I.: Early transcriptional response of Saccharomyces cerevisiae to stress imposed by the herbicide 2, 4-dichlorophenoxyacetic acid. FEMS Yeast Research 6(2), 230–248 (2006)
25. Blaiseau, P.L., Lesuisse, E., Camadro, J.M.: Aft2p, a novel iron-regulated transcription activator that modulates, with Aft1p, intracellular iron use and resistance to oxidative stress in yeast. Journal of Biological Chemistry 276(36), 34221–34226 (2001)
26. Harbison, C.T., Gordon, D.B., Lee, T.I., Rinaldi, N.J., Macisaac, K.D., Danford, T.W., Hannett1, N.M., Tagne, J.-B., Reynolds, D.B., Yoo, J., Jennings, E.G., Zeitlinger, J., Pokholok, D.K., Kellis, M., Rolfe, P.A., Takusagawa, K.T., Lander, E.S., Gifford, D.K., Fraenkel, E., Young, R.A.: Transcriptional regulatory code of a eukaryotic genome. Nature 431(7004), 99–104 (2004)

Mean Squared Residue Based Biclustering Algorithms

Stefan Gremalschi* and Gulsah Altun*

Department of Computer Science,
Georgia State University,
Atlanta, GA 30303
{stefan,gulsah}@cs.gsu.edu

Abstract. The availability of large microarray data has brought along many challenges for biological data mining. Following Cheng and Church [4], many different biclustering methods have been widely used to find appropriate subsets of experimental conditions. Still no paper directly optimizes or bounds the Mean Squared Residue (MSR) originally suggested by Cheng and Church. Their algorithm, for a given expression matrix A and an upper bound on MSR, finds k almost non overlapping biclusters whose sizes are not predefined thus making it difficult to compare with other methods.

In this paper, we propose two new Mean Squared Residue (MSR) based biclustering methods. The first method is a dual biclustering algorithm which finds $(k \times l)$-bicluster with MSR using a greedy approach. The second method combines dual biclustering algorithm with quadratic programming. The dual biclustering algorithm reduces the size of the matrix, so that the quadratic program can find an optimal bicluster reasonably fast. We control bicluster overlapping by changing the penalty for reusing cells in biclusters. The average MSR in [4] biclusterings for yeast is almost the same as for the proposed dual biclustering while the median MSR is 1.5 times larger thus implying that the quadratic program finds much better smaller biclusters.

1 Introduction

The availability of large microarray data has brought along many challenges for biological data mining because measurements are taken in multiple biological conditions which are not related to the biological questions being asked. To overcome this problem, a method called biclustering has been widely used to find appropriate subsets of experimental conditions and many algorithms have been proposed [1],[5],[7],[10],[12],[13] and [14].

Gene expression data generated by DNA chips and other microarray techniques are often presented as matrices of expression levels of genes under different conditions (including environments, individuals, and tissues) [2]. One of the usual goals in expression data analysis is to group genes according to their expression under multiple conditions, or to group conditions based on the expression of a number of genes. This may lead to discovery of regulatory patterns or condition similarities. The current practice is often the application of some agglomerative or divisive clustering algorithm that partitions

* Partially supported by GSU Molecular Basis of Disease Fellowship.

I. Măndoiu, R. Sunderraman, and A. Zelikovsky (Eds.): ISBRA 2008, LNBI 4983, pp. 232–243, 2008.

the genes or conditions into mutually exclusive groups or hierarchies. The basis for clustering is often the similarity between genes or conditions as a function of the rows or columns in the expression matrix.

Biclustering was introduced by Cheng and Church [4] and their algorithm is based on a simple uniformity goal which is the mean squared residue. However, this algorithm tends to generate large biclusters that often represent gene groups with unchanged expression levels. Therefore interesting patterns in terms of co-regulation are not necessarily contained [7].

To overcome this problem, we propose two new MSR based biclustering methods in this paper. The first method is a dual biclustering algorithm which finds $(k \times l)$-bicluster with MSR using a greedy approach. The second method combines dual biclustering algorithm with quadratic programming (QP). The dual biclustering algorithm reduces the size of the matrix, so that the quadratic program can find optimal bicluster reasonably fast. We control bicluster overlapping by changing the penalty for reusing cells in biclusters. The average MSR in [4] biclusterings for yeast is almost the same as for the proposed dual biclustering while the median MSR is 1.5 times larger thus implying that the quadratic program finds much better smaller biclusters, which are functionally enriched and indicate a strong correspondence with known pathways.

The reminder of this paper is organized as follows. Section 2 gives the formal definition of mean squared residue. Cheng and Church's algorithm [4] is briefly described in Section 3. Section 4 defines dual biclustering problem, describes the algorithm and bicluster overlapping control method. Section 5 defines Dual Biclustering as an optimization problem and describes the quadratic program. The analysis and validation of experimental study is given in Section 6. Finally, we draw conclusions in Section 7.

2 Mean Squared Residue

Mean squared residue problem has been defined before by Cheng and Church [4] and Zhou and Khokhar [14]. In this paper, we use the same terminology as in [14]. Our input is an $(N \times M)$-data matrix A, with R rows and C columns, where a cell a_{ij} is a real value that represents the expression level of gene i(row i), under condition j(column j). Matrix A is defined by its set of rows, $R = \{r_1,\ r_2,\ ...,\ r_N\}$ and its set of columns $C = \{c_1,\ c_2,\ ...,\ c_M\}$.

Given a matrix, biclustering finds sub-matrices, that are subgroups of rows (genes) and subgroups of columns, where the genes exhibit highly correlated behavior for every condition. Given a data matrix A, the goal is to find a set of biclusters such that each bicluster exhibits some similar characteristic. Let $A_{IJ} = (I,\ J)$ represent a submatrix of A ($I \in R$ and $J \in C$). A_{IJ} contains only the elements a_{ij} belonging to the submatrix with set of rows I and set of columns J. A bicluster $A_{IJ} = (I,\ J)$ can be defined as a k by l sub-matrix of the data matrix where k and l are the number of rows and the number of columns in the submatrix A_{IJ}. The concept of bicluster was introduced by [4] to find correlated subsets of genes and a subset of conditions.

Let a_{iJ} denote the mean of the i-th row of the bicluster $(I,\ J)$, a_{Ij} the mean of the j-th column of $(I,\ J)$, and a_{IJ} the mean of all the elements in the bicluster. As given in [4], more formally,

$$a_{iJ} = \frac{1}{|J|}\sum_{j\in J} a_{ij},\ a_{Ij} = \frac{1}{|I|}\sum_{i\in I} a_{ij}\ and\ a_{IJ} = \frac{1}{|I||J|}\sum_{i\in I, j\in J} a_{ij}$$

According to [4], the residue of an element a_{ij} in a submatrix A_{IJ} equals

$$r_{ij} = (a_{ij} - a_{iJ} - a_{Ij} + a_{IJ})$$

The difference between the actual value of a_{ij} and its expected value predicted from its row, column, and bicluster mean is given by the residue of an element. It also reveals its degree of coherence with the other entries of the bicluster it belongs to. The quality of a bicluster can be evaluated by computing the mean squared residue H, i.e. the sum of all the squared residues of its elements[4]:

$$H(I,J) = \frac{1}{|I||J|}\sum_{i\in I, j\in J} (a_{ij} - a_{iJ} - a_{Ij} + a_{IJ})^2$$

A submatrix A_{IJ} is called a $\delta - bicluster$ if $H(I,J) \leq \delta$ for some given threshold $\delta \geq 0$.

In general, we can formulate biclustering problem bilaterally – maximize the size (area) of the biclusters and minimize MSR. But, these two objectives above contradict each other because smaller biclusters have smaller MSR and vice versa. Therefore, there are two optimization problem formulations. Cheng and church considered the following formulation: Maximize the bicluster size (area) subject to an upper bound on MSR. In section 4, we consider the dual formulation minimize MSR subject to lower bound on size (area) of biclusters.

3 Cheng and Church's Algorithm

In this section, we briefly describe Cheng and Church's algorithm [4][9]. The algorithm proposed by Cheng and Church in [4] is based on a simple uniformity goal which is the mean squared residue [9]. It also uses a greedy approach to find one bicluster that is combined iteratively to find more biclusters. The biclustering algorithm searches for a δ-bicluster assuming that the parameter δ was chosen appropriately to avoid random signal identification. The optimization problem of identifying the the largest δ-bicluster is NP hard. Thus, a heuristics is needed for finding a large δ-bicluster in reasonable time.

A naive greedy algorithm for finding δ-bicluster starts with the given data matrix and in a brute force manner tries all single rows (columns) addition (deletion), applying the best operation if it improves the score and terminates when no such operation exists or when the bicluster score is below a certain δ threshold value. However, for large matrices this calculation is very time consuming. To accelerate steps in the greedy algorithm, Cheng and Church proposed a method uses the structure of the mean residue. The underlying idea is based on lemma 1 [4]:

Lemma 1. *The set of rows (columns) that can be completely or partially removed with the net effect of decreasing the mean residue score of a bicluster A_{IJ} is:*

$$R = \{i \in I; \frac{1}{|J|} \sum_{j \in J} RS_{IJ}(i,j) > H(I,J)\}$$

Lemma 1 states that any row (column) can be removed if their average contribution to the score is grater than its relative share. This argument gives rise to the following greedy algorithm that iteratively removes rows (columns) with the maximal average residue score (Figure 1)[9].

Lemma 2. *The set of rows (columns) that can be completely or partially added with the net effect of decreasing the mean squared residue score of a bicluster A_{IJ} is (Figure 2) [9]:*

$$R = \{i \notin I; \frac{1}{|J|} \sum_{j \in J} RS_{IJ}(i,j) \leq H(I,J)\}$$

Input: Expression matrix A on genes S, conditions C and a parameter δ.
Output: $A_{I,J}$ a δ-bicluster.

Initialize: $I = S,\ J = C$.
Iteration:

1. Calculate a_{iJ}, a_{Ij} and $H(I,J)$. If $H(I,J) < \delta$ output I, J.
2. For each row calculate $d(i) = \frac{1}{|J|} \sum_{j \in J} RS_{IJ}(i,j)$
3. For each column calculate $e(j) = \frac{1}{|I|} \sum_{i \in I} RS_{IJ}(i,j)$
4. Take the best row or column and remove it from I or J.

Fig. 1. Single node deletion algorithm

Input: Expression matrix A, parameter δ, I, J specifying a δ-bicluster.
Output: $A_{I',J'}$ a δ-bicluster with $I' \subseteq I$ and $J' \subseteq J$.

Iteration:

1. Calculate a_{iJ}, a_{Ij} and $H(I,J)$.
2. Add the columns with $\frac{1}{|I|} \sum_{i \in I} RS_{IJ}(i,j) \leq H(I,J)$
3. Calculate a_{iJ}, a_{Ij} and $H(I,J)$.
4. Add the rows with $\frac{1}{|J|} \sum_{j \in J} RS_{IJ}(i,j) \leq H(I,J)$
5. If nothing was added, halt.

Fig. 2. Node addition algorithm

Cheng and Church also suggest two improvements to their basic deletion/addition algorithm. The first improvement is for large data sets where multiple node deletion can be done by removing at each deletion iteration all rows (columns) for which $d(i) > \alpha H(I,J)$ for some choice of α. The second improvement is to add inverse rows to the

matrix which makes it easier to find biclusters which contains co-regulation and inverse co-regulation. Cheng and Church's algorithm uses the δ-bicluster algorithm as a subroutine and repeatedly applies it to the matrix. In this method, one problem would be to find the same bicluster over and over again. However, in Cheng and Church's algorithm the discovered bicluster is masked by replacing the values of its submatrix with random values. The general biclustering scheme is outlined in Figure 3 [9].

Input: Expression matrix A, parameter δ and k-the number of biclusters to report.
Output: k δ-biclusters in matrix A.

Iteration:

1. Apply multiple node deletion on A giving I' and J'.
2. Apply node addition on I' and J' giving $I^{"}$ and $J^{"}$.
3. Store $I^{"}$, $J^{"}$ and replace $A_{I^{"}J^{"}}$ values by random numbers.

Fig. 3. Cheng and Church's biclustering algorithm

4 Dual Biclustering

In this section, we first define dual biclustering problem, describe the algorithm and bicluster overlapping control method.

As we mentioned in Section 2, we can formulate biclustering problem bilaterally – maximize the size (area) of the biclusters and minimize MSR. These two objectives above contradict each other because smaller biclusters have smaller MSR and vice versa. We formulate the dual biclustering problem as follows: given expression matrix A, find $k \times l$ bicluster with the smallest mean squared residue H. For a set of biclusters, we have:

Given: matrix $A_{n\times m}$, set of bicluster sizes S, total overlapping V.
Find: $|S|$ biclusters with total overlapping at most V and total minimum sum of scores H.

4.1 Dual Biclustering Algorithm

The greedy algorithm for finding a bicluster may start with the entire matrix and at each step try all single rows (columns) addition (deletion), applying the best operation if it improves the score and terminating when it reaches the bicluster size $k \times l$. The output bicluster will have the smaller MSR for the given size. Like in [4], the algorithm uses the structure of the mean residue score to enable faster greedy steps: for a given threshold α, at each deletion iteration all rows (columns) for which $d(i) > \alpha H(I, J)$ are removed. Also, the algorithm implements the addition of inverse rows to the matrix, allowing the identification of the biclusters which contains co-regulation and inverse co-regulation.

This algorithm is used as a subroutine and repeatedly applied to the matrix. We are using bicluster overlapping control (BOC) to avoid finding the same bicluster over and over again. The penalty is applied for using the cells present in biclusters found before.

Input: Expression matrix A on genes n, conditions m and bicluster size (k, l).
Output: Bicluster $A_{I,J}$ with the smallest MSR.

Initialize: $I = n,\ J = m, \forall w_(i,j) = 0, i \in n, j \in m.$
Iteration:

1. Calculate a_{iJ}, a_{Ij} and $H(I, J)$. If $|I| = k,\ |J| = l$ output I, J.
2. For each row calculate $d(i) = \frac{1}{|J|} \sum_{j \in J} RS'_{IJ}(i, j)$
3. For each column calculate $e(j) = \frac{1}{|I|} \sum_{i \in I} RS'_{IJ}(i, j)$
4. Take the best row or column and remove it from I or J.

Fig. 4. Single node deletion algorithm

Input: Expression matrix A and bicluster size (k, l).
Output: Bicluster $A_{I',J'}$ with $I' \subseteq I$ and $J' \subseteq J$.

Iteration:

1. Calculate a_{iJ}, a_{Ij} and $H(I, J)$.
2. Add the columns with $\frac{1}{|I|} \sum_{i \in I} RS'_{IJ}(i, j) \leq H(I, J)$
3. Calculate a_{iJ}, a_{Ij} and $H(I, J)$.
4. Add the rows with $\frac{1}{|J|} \sum_{j \in J} RS'_{IJ}(i, j) \leq H(I, J)$
5. If nothing was added or $|I'| = k$, $|J'| = l$, halt.

Fig. 5. Single node addition algorithm

By using BOC, we can preserve the original data from losing information it carries because we do not mask biclusters with random numbers. The general biclustering scheme is outlined in Figure 6.

4.2 Bicluster Overlapping Control

It was noted in [4] that we need to find almost non overlapping biclusters. Therefore we introduce the measure of bicluster overlapping V which is one's complement of the ratio of number of distinct cells used in all found biclusters divided by the total area of all biclusters. In order to control the bicluster overlapping, we remove columns and rows based on the number of cells that have been used in previously extracted biclusters. We can achieve the given bicluster overlapping by giving more or less penalty for reusing cells.

Let $A_{n \times m}$ be the input matrix, $W_{n \times m}$ the weight matrix where $w_{ij} \in \{0, 1\}$ and a bicluster A_{IJ}. The weight matrix $W_{n \times m}$ is initialized to 0. When a bicluster A_{IJ} is found, the weight matrix elements W_{IJ} are set to 1. The average row (column) contribution to the mean squared residue is given by the following formula:

$$RS'_{IJ}(i, j) = \frac{1}{|J|} \sum_{j \in J} (a_{ij} - a_{iJ} - a_{Ij} + a_{IJ})^2 + w_{ij} \vartheta H(I, J)$$

where ϑ is an overlapping parameter. If a cell is used before in some bicluster, then $w_{ij} = 1$, which enables the penalty for reusing this cell.

Input: Expression matrix A, parameter α and a set of bicluster sizes S.
Output: $|S|$ biclusters in matrix A.

Iteration:

1. $\forall w_(i,j) = 0, i \in n, j \in m.$
2. while S not empty do
3. $(k,l) = \ get\ first\ element\ from\ S$
4. $S = S - \{(k,l)\}$
5. Apply multiple node deletion on A giving (k,l).
6. Apply node addition on A' giving (k,l).
7. Store $A^{"}$ and update W.
8. end.

Fig. 6. Dual biclustering algorithm

5 Mean Squared Residue Minimization Via Quadratic Program

We first define the Dual Biclustering as an optimization problem [6], [3]. Then, we define the quadratic program for biclustering and show how to write its objective and constraints. We conclude with QP results interpretation.

Although greedy algorithms run fast and give a solution to the problem, it happens that in many cases this solution is not optimal. Quadratic Program (QP) is one of the optimization methods and is known for always providing optimal solution for the problem it solves. It has an objective which is a quadratic function of the decision variables, and constraints which are all linear functions of the variables.

We give the the Dual Biclustering formulation as an optimization problem: for a given matrix $A_{n\times m}$, find the bicluster with bounded size (area) $k \times l$ with minimal mean squared residue.

It can be easily seen that if MSR has to be defined as QP objective, it will be of a cubic form. Since QP's objective can be contain only squared variables, the following constraint needs to be satisfied: define QP objective in such a way that only quadratic variables are present. To meet this requirement, we simulated variable multiplication by addition. Next subsection describes multiplication simulation.

5.1 Linear Representation of Multiplication

For every element a_{ij} from matrix A we introduce a variable x_{ij}. This variable equals to 1 if and only if both $row_i \in I$ and $column_j \in J$, otherwise it equals to 0. In other words, $x_{ij} = row_i \cdot column_j$. Assuming that $x_{ij},\ row_i\ and column_j$ are binary variables, i.e. can be only 0 or 1, we define a rule that substitutes the multiplication with addition:

$$x_{ij} \geq row_i + column_j - 1$$
$$x_{ij} \leq row_i$$
$$x_{ij} \leq column_j$$

Indeed, if $row_i = 0$ or $column_j = 0$ then the second and the third inequality guarantee that $x_{ij} = 0$. If both $row_i = 1$ and $column_j = 1$ then all three inequalities guarantee that $x_{ij} = 1$.

All variable multiplications can be simulated by addition by using similar constraints. For that, we need to normalize the original matrix $A_{n\times m}$ so all its entries are from $[0, 1]$ interval. Data normalization is made as follows:

$$a_{ij}{}' = \frac{1}{2} + \frac{a_{ij} - min(A_{n\times m})}{2(max(A_{n\times m}) - min(A_{n\times m}))}$$

Additional inverted rows are added to the normalized matrix. Quadratic program will search for inverted gene expression profiles like dual algorithm does. The final matrix $A'_{2n\times m}$ will have twice more rows than the original matrix $A_{n\times m}$. Section 5.1 presents the quadratic program for biclustering.

5.2 Integer Quadratic Program

For a given normalized matrix $A_{n\times m}$ and bicluster size $k \times l$, the Integer Quadratic Program is defined as follows:

Objective

$Minimize : \frac{1}{|I||J|} \sum_{i\in n, j\in m} (residue_{ij})^2$

Subject to

$I = k$
$J = l$
$residue_{ij} = a_{ij}x_{ij} - a_{iJ}x_{ij} - a_{Ij}x_{ij} + a_{IJ}x_{ij}$
$a_{iJ} = \frac{1}{|J|} \sum_{j\in m} a_{ij},\ a_{Ij} = \frac{1}{|I|} \sum_{i\in n} a_{ij}\ and\ a_{IJ} = \frac{1}{|I||J|} \sum_{i\in n,\ j\in m} a_{ij}$
$x_{ij} \geq row_i + column_j - 1$
$x_{ij} \leq row_i$
$x_{ij} \leq column_j$
$\sum_{i\in n} row_i = k$
$\sum_{j\in m} column_j = l$
$x_{ij},\ row_i,\ column_j \in \{0,\ 1\}$

End

The QP is used as a subroutine and repeatedly applied to the matrix. For each bicluster size, we generate a separate QP. In order to avoid finding the same bicluster over and over again, the discovered bicluster is masked by replacing the values of its submatrix with random values.

5.3 Rounding of Fractional Relaxation

The integer QP is too slow and its not scalable enough. Fractional relaxation of QP is much faster [8]. If we allow the variables x_{ij}, row_i, and $column_j$ to take values from $[0, 1]$ interval, we will obtain a fractional quadratic program. This chance can speed

up the time required by solver to give the solution to the QP. The drawback of the fractional QP is how to interpret the solution. This section gives a description of QP results interpretation.

The output values for variables of the relaxed quadratic program belong to the $(0, 1)$ - interval, which makes the selection decision not obvious. We propose two ways of interpreting results from quadratic program: greedy rounding and random interval rounding.

Greedy Rounding method sorts values of all variables obtained in descending order. It returns the first k rows and l columns. The assumption in this method is that if a node has a value close or equal to 1, then there is a high probability that this node belongs to the final solution set.

In random interval rounding selection, we build an interval for each variable from the output file of the quadratic program: higher the value, larger the interval. The node is selected by generating a random number which is checked in which interval it falls. When all k rows and l columns are selected, the algorithm computes the mean squared residue. This procedure is repeated 100 times and the final solution will contain the set of nodes with the smallest MSR value.

5.4 Combining Dual Biclustering with Rounded QP

In this section, we propose a combined Dual Biclustering and Rounded QP algorithm. The main idea is to reduce the instance size to speed up the QP. First, we apply Dual Algorithm to input matrix A to reduce the instance size. New size is specified by two parameters: $ratio_k$ and $ratio_l$. Then we run Rounded QP on the output obtained from Dual Biclustering algorithm. This combination improves the running time of the QP and increases the quality of the final bicluster since an optimization method is applied. The general algorithm scheme is outlined in Figure 7.

6 Experimental Results

In this section, we analyse results obtained from Dual Biclustering and Quadratic Program algorithms. We describe comparison criteria, define the swap rule model and analyze the p value of the biclusters.

We tested our biclustering algorithms on data from [11] and compared our results with [4]. For a fair comparison, we used bicluster sizes published in [4]. The average mean squared residue of [4] biclusters for yeast is 204.29 with overlap 18%, while our method finds biclusters with average MSR value equal to 205.76 with overlap 17%. Medians are 196.30 and 123.27, respectively. Thus, implying that our algorithm finds much better smaller biclusters. In case of QP, it found 45 from 100 biclusters with much smaller MSR than in [4]. Most of biclusters where QP won have all l columns. Results are summarized in Figure 8.

According to [7], Cheng and Church's algorithm tends to generate large biclusters that often represent gene groups with unchanged expression levels and therefore not necessarily contain interesting patterns in terms of, e.g. co-regulation. Instead, small biclusters are functionally enriched and indicate a strong correspondence with known pathways. We have selected a set containing 66 biclusters with sizes not exceeding 400 rows and 17 columns. The results are summarized in Figure 9.

Input: Expression matrix A, parameters α, $ratio_k$, $ratio_l$ and a set of bicluster sizes S.
Output: $|S|$ biclusters in matrix A.

1. while S not empty do
2. $(k, l) = \ get\ first\ element\ from\ S$
3. $S = S - \{(k, l)\}$
4. $k' = k \cdot ratio_k$
5. $l' = l \cdot ratio_l$
6. Apply multiple node deletion on A giving (k', l').
7. Apply node addition on A' giving (k', l').
8. Update W.
9. Run QP on $A^{"}$ giving (k', l').
10. Round Fractional Relaxation and store $A^{"}$.
11. end.

Fig. 7. Combined Dual Biclustering with Rounded QP algorithm

	Algorithms			
	Cheng and Church	Dual Biclustering		Dual and QP
OC parameter	n/a	1.6	1.8	1.8
Overlapping	39945	39577	40548	41119
Average MSR	204.29323	190.82	205.77	171.19
(%)	100	93.4	100.72	83.79
Median MSR	196.3095	117.96	123.27	102.56
(%)	100	60.1	62.79	52.24

Fig. 8. Results from running on [11] dataset and 100 biclusters published by [4]

	Algorithms			
	Cheng and Church	Dual Biclustering		Dual and QP
OC parameter	n/a	1.6	1.8	1.8
Average MSR	208.81	170.32	182.96	157.77
(%)	100	81.57	87.62	75.55
Median MSR	205.15	100.1	101.13	84.12
(%)	100	48.78	49.3	41

Fig. 9. Results from running on [11] dataset and 85 biclusters published by [4], with sizes not exceeding 400 rows and 17 columns

We measure the statistical significance of biclusters obtained by our algorithms using p value. P value is computed by running Dual Problem algorithm on 100 random generated input data sets. The random data is obtained from matrix A by randomly selecting two cells in the matrix (a_{ij}, d_{kl}) and taking their diagonal elements (b_{kj}, c_{il}). If $a_{ij} > b_{kj}$ and $c_{il} < d_{kl}$, algorithm swaps a_{ij} with c_{il} and b_{kj} with d_{kl}. It is called a hit. If not, two elements a_{ij} and d_{kl} are randomly chosen again. The matrix is considered randomized if there are $\frac{nm}{2}$ hits. In our case, p value is smaller than 0.001, which indicates that the results are not random and are statistically significant.

7 Conclusions

Biclustering was introduced by [4] and their algorithm is based on a simple uniformity goal which is the mean squared residue. But this algorithm tends to generate large biclusters that often represent gene groups with unchanged expression levels and therefore not necessarily contain interesting patterns in terms of co-regulation [7].

To overcome this problem, we propose two new MSR based biclustering methods. The first method is a dual biclustering algorithm which finds $(k \times l)$-bicluster with MSR using a greedy approach. The second method combines dual biclustering algorithm with quadratic programming. The dual biclustering algorithm reduces the size of the matrix, so that the quadratic program can find optimal bicluster reasonably fast. Proposed algorithms can find smaller size biclusters with MSR almost 3 times smaller than MSR values reported in [4]. According to [7], this is a great advantage because small biclusters indicate a strong correspondence with known biclusters. The average MSR for all biclusters in [4] is almost the same as for the proposed dual biclustering while the median MSR is 1.5 times larger thus implying that proposed algorithms find much better smaller biclusters.

We also have introduced a method for controlling bicluster overlapping, which enables a fair comparison between different biclustering algorithms proposed in literature.

References

1. Angiulli, F., Pizzuti, C.: Gene Expression Biclustering using Random Walk Strategies. In: Tjoa, A.M., Trujillo, J. (eds.) DaWaK 2005. LNCS, vol. 3589, Springer, Heidelberg (2005)
2. Baldi, P., Hatfield, G.W.: DNA Microarrays and Gene Expression. From Experiments to Data Analysis and Modelling. Cambridge Univ. Press, Cambridge (2002)
3. Bertsimas, D., Tsitsiklis, J.: Introduction to Linear Optimization, Athena Scientific (1997)
4. Cheng, Y., Church, G.M.: Biclustering of Expression Data. In: Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology, pp. 93–103. AAAI Press, Menlo Park (2000)
5. Madeira, S.C., Oliveira, A.L.: Biclustering Algorithms for Biological Data Analysis: A Survey. IEEE Transactions on Computational Biology and Bioinformatics 1(1), 24–45 (2004)
6. Papadimitriou, C.H., Steiglitz, K.: Combinatorial optimization: algorithms and complexity. Prentice-Hall, Inc, Upper Saddle River, NJ (1982)
7. Prelic, A., Bleuler, S., Zimmermann, P., Wille, A., Bühlmann, P., Gruissem, W., Hennig, L., Thiele, L., Zitzle, E.: A systematic comparison and evaluation of biclustering methods for gene expression data. Bioinformatics 22(9), 1122–1129 (2006)

8. Ravikumar, P., Lafferty, J.: Quadratic programming relaxations for metric labeling and Markov random field MAP estimation. In: Proceedings of the 23rd international conference on Machine learning, pp. 737–744 (2006)
9. Shamir, R.: Lecture notes, http://www.cs.tau.ac.il/~rshamir/ge/05/scribes/lec04.pdf
10. Tanay, A., Sharan, R., Shamir, R.: Discovering Statistically Significant Biclusters in Gene Expression Data. Bioinformatics 18, 136–144 (2002)
11. Tavazoie, S., Hughes, J.D., Campbell, M.J., Cho, R.J., Church, G.M.: Systematic determination of genetic network architecture. Nature Genetics 22, 281–285 (1999)
12. Yang, J., Wang, H., Wang, W., Yu, P.: Enhanced biclustering on gene expression data. In: Proceedings of the 3rd IEEE Conference on Bioinformatics and Bioengineering (BIBE), pp. 321–327 (2003)
13. Zhang, Y., Zha, H., Chu, C.H.: A time-series biclustering algorithm for revealing co-regulated genes. In: Proc. Int. Symp. Infoation and Technology: Coding and Computing (ITCC 2005), Las Vegas, USA, pp. 32–37 (2005)
14. Zhou, J., Khokhar, A.A.: ParRescue: Scalable Parallel Algorithm and Implementation for Biclustering over Large Distributed Datasets. In: Proc. of the 26th IEEE International Conference on Distributed Computing Systems, ICDCS (2006)

Sparse Decomposition of Gene Expression Data to Infer Transcriptional Modules Guided by Motif Information

Ting Gong[1], Jianhua Xuan[1,*], Li Chen[1], Rebecca B. Riggins[2], Yue Wang[1], Eric P. Hoffman[3], and Robert Clarke[2]

[1] Department of Electrical and Computer Engineering, Virginia Polytechnic Institute and State University, Arlington, VA 22203, USA
{tinggong, xuan, lchen06, yuewang}@vt.edu
[2] Departments of Oncology and Physiology & Biophysics, Georgetown University School of Medicine, Washington, DC 20057, USA
{rbr7, clarker}@georgetown.edu
[3] Research Center for Genetic Medicine, Children's National Medical Center, Washington, DC 20010, USA
ehoffman@cnmcresearch.org

Abstract. An important topic in computational biology is to identify transcriptional modules through sequence analysis and gene expression profiling. A transcriptional module is formed by a group of genes under control of one or several transcription factors (TFs) that bind to *cis*-regulatory elements in the promoter regions of those genes. In this paper, we develop an integrative approach, namely motif-guided sparse decomposition (mSD), to uncover transcriptional modules by combining motif information and gene expression data. The method exploits the interplay of co-expression and co-regulation to find regulated gene patterns guided by TF binding information. Specifically, a motif-guided clustering method is first developed to estimate transcription factor binding activities (TFBAs); sparse component analysis is then followed to further identify TFs' target genes. The experimental results show that the mSD approach can successfully help uncover condition-specific transcriptional modules that may have important implications in endocrine therapy of breast cancer.

Keywords: Motif analysis, sparse component analysis, transcriptional modules, gene regulatory networks, estrogen receptor binding.

1 Introduction

Exacting clear and coherent hypothesis from genome-wide expression data remains a challenging problem. Many computational tools have been developed to facilitate the identification of differentially expressed genes and their significance in a variety of experimental designs [1]. Recent research has discovered that the production of transcripts of a given gene is governed by a complex combinational interplay of

* Corresponding author.

I. Măndoiu, R. Sunderraman, and A. Zelikovsky (Eds.): ISBRA 2008, LNBI 4983, pp. 244–255, 2008.

cis-regulatory elements (henceforth referred to as motifs) [2]. Associated transcription factors (TFs) act alone or in combination on target promoters to control gene expression. Transcription factor's regulatory activity is controlled by higher level cellular functions, such as signaling pathways to reflect cellular physiology and environment. Therefore many efforts have been made in exploring the clustering of genes into transcriptional modules - a collection of genes under (perhaps combinational) control of a set of transcription factors that bind to regulatory elements in the promoter regions for those genes.

One such strategy is to use motif discovery algorithms to search for recurring patterns in a given set of related sequences such as AlignACE [3] or to search for known binding sites based on a predefined library of all previously characterized motifs or position weight matrices (PWMs). Unfortunately, using a strictly bioinformatics-based approach to identify target genes of transcription factors is still extremely challenging because most transcription factor binding sites (TFBSs) are degenerate sequences that occur quite frequently in the mammalian genome [4]. Other strategies have been used in combination with expression data and ChIP-on-chip data. Although many tools such as MarsMotifs [5] have aided experimental biologists in the discovery of regulatory information, a large false-positive prediction rate is still a major problem.

A computational approach, network component analysis (NCA), has been recently developed to reconstruct the profiles of TFs faithfully [6]. However it relies heavily on the availability of connectivity information from ChIP-on-chip data. Therefore, NCA scheme is not applicable to many biological studies where adequate connectivity information is unavailable, due to lack of complete ChIP-on-chip data. It is often the case that both the connectivity structure of the TFs and their targets and the activity profiles of the TFs have to be reconstructed.

In this paper, we focus on the problem of transcriptional module identification, which essentially requires finding sets of transcription factor binding sites that co-occur in promoter regions of genes with a common expression pattern. In order to learn the membership of transcriptional modules, we propose to combine motif information and expression data in a novel way: (1) using motif information to guide finding regulated gene patterns, and (2) using a sparse component analysis (SCA) method [7] to further decompose the regulated gene patterns hence to recover the TF-gene connectivity information. This two-step approach will be termed as motif-guided sparse decomposition (mSD) method in this paper. We have applied the mSD approach to a breast cancer data set to identify estrogen-dependent transcriptional modules. The experimental results demonstrated that with the help of the proposed mSD method, we can successfully identify condition-specific transcriptional modules that may have important implications in anti-estrogen therapy of breast cancer.

The paper is organized as follows. In Section 2, we give a detailed description of our motif-guided sparse decomposition (mSD) method for transcriptional module identification. In Section 3, we present the experimental results on an estrogen-dependent profiling study of breast cancer, focusing on condition-specific transcriptional modules recovered by our mSD approach. Finally, in Section 4, we give the conclusion of this paper.

2 Method

In this paper, we make a simplified yet biologically plausible assumption that the overlap of influences of TFBSs is additive:

$$X_{sg} = \sum_t X(s, g|t), \tag{1}$$

where X_{sg} is defined as the logarithm of the expression ratio of gene g between the current data sample s (or a particular time point) and the control vehicle, while $X(s, g|t)$ is the expression level of g in s due to transcription factor binding site t. We also assume that $X(s, g|t)$ is multiplicatively decomposable into the activity level (A_{st}) of TFBS t in sample s and the regulation strength (S_{tg}) of t onto gene g:

$$X(s, g|t) = A_{st} \bullet S_{tg}. \tag{2}$$

Combing (1) and (2) leads to a reformulation of canonical matrix form as:

$$X_{sg} = \sum_t A_{st} \bullet S_{tg}. \tag{3}$$

The log ratios of gene expression $X \in \Re^{m \times N}, (N \gg 1)$ are expressed as a linear combination of log ratios of TFBS activity (TFBSA) level ($A \in \Re^{m \times n}$) weighted by their regulation strength ($S \in \Re^{n \times N}$). Note that m is the number of samples, N the number of genes and n the number of TFBSs.

We notice that the number of TFBSs is much smaller than the number of transcribed genes and most genes are regulated only by a small number of TFBSs. Hence, the matrix S that describes the connections between the TFBSs and their regulated genes is sparse. Further, we should point out that we do not perform motif discovery as part of our learning procedure, but rather assume that we have a list of motifs for putative transcription factor binding sites by searching a database of regulatory elements such as TRANSFAC [8]. Since we are using motif profiles with respect to a known set of motifs as a source of data, usually the number of TFBSs (n) or motifs is greater than our sample number (m). That is to say, generally, $n > m$, which is equivalent to say that Equation (3) is an underdetermined linear system (ULS). Considering the sparse property of S and ULS property of Equation (3), we propose to employ a sparse component analysis (SCA) method [7] to estimate S for regulatory module identification. However, the SCA method requires having the TFBSA matrix (A) known beforehand, which in our case needs to be estimated as well.

To fully solve this problem, we propose to develop a motif-guided sparse decomposition (mSD) method for transcriptional module identification. The mSD method is composed of two steps described as follows. In the first step, we will develop a motif-guided clustering algorithm to find regulated gene expression patterns to facilitate the estimation of A; in the second step, we will employ the SCA method to iteratively detect active TFBSs and estimate regulation strength matrix S. In the following subsections, we present the two algorithms in our proposed mSD approach: (1) TFBSA matrix (A) identification algorithm and (2) regulation strength matrix (S) recovery algorithm.

2.1 Inferring TFBS Activity Matrix by Motif-Guided Clustering

In order to reliably estimate the TFBSA A matrix from Equation (3), the sparsity property of the regulation strength matrix S is very important. In fact, the following theorem is the key to obtaining a reliable estimation of A [7].

Theorem 1: (Identifiability Conditions - Locally Very Sparse Representation): Assume that the number of TFBSs is unknown and the following: 1) Each TFBS has at least two *strictly well-grounded points*, which means that for each index $i = 1, \cdots, n$, there are at least two columns of *S: S(:, j_1)* and *S(:, j_2)* which have nonzero elements only in position i (so each TFBS is uniquely present at least twice); 2) $X(:,k) \neq cX(:,q)$ for any $c \in \Re$, any $k = 1, \cdots, N$ and any $q = 1, \cdots, N$, $k \neq q$ for which $S(:, k)$ has more than one nonzero element. Then A is uniquely determined by X except for left multiplication with a permutation and scaling matrix.

For proofs of the theorem we refer to [7]. An algorithm to obtain A can be summarized as follows [7]:

1) Remove all zero columns of X (if any) and obtain a matrix $X_1 \in \Re^{m \times N_1}$.
2) Normalize the columns $\boldsymbol{x}_i$, $i = 1, \cdots, N_1$ of X_1: $\boldsymbol{y}_i = \dfrac{\boldsymbol{x}_i}{\|\boldsymbol{x}_i\|}$. Multiply each column $\boldsymbol{y}_i$ by -1 if the first element of $\boldsymbol{y}_i$ is negative.
3) Cluster $\boldsymbol{y}_i$, $i = 1, \cdots, N_1$ in l groups: $G_1, \ldots G_l$ such that for any $j = 1, \cdots, l$, $\|\boldsymbol{x} - \boldsymbol{y}\| < \varepsilon, \forall \boldsymbol{x}, \boldsymbol{y} \in G_j$ and $\|\boldsymbol{x} - \boldsymbol{y}\| \geq \varepsilon$ for any $\boldsymbol{x}$, $\boldsymbol{y}$ belonging to different groups.
4) Chose any $\boldsymbol{y}_i \in G_i$ and put $\boldsymbol{a}_i = \boldsymbol{y}_i$.

In the above algorithm, Step (3) requires a clustering method to find representative columns of X to estimate A. There have been many clustering techniques proposed for clustering gene expression data, such as k-means clustering [9] and self-organizing maps [10], which are designed to find gene expression patterns by grouping the genes with similar expression profiles. Very recently, an affinity propagation (AP) algorithm has been proposed for data clustering that showing an improved performance [11]. Based on an *ad hoc* pairwise similarity function between data points, AP seeks to identify each cluster by one of its elements, the so-called *exemplar*. AP takes as input a collection of real-valued similarities between data points, where the similarity $s(i, k)$ indicates how well the data point with index k is suited to be the exemplar for data point i. When the goal is to minimize squared error, each similarity is set to a negative squared error (Euclidean distance): For points x_i and x_k, $s(i,k) = -\|x_i - x_k\|^2$ [11].

However, direct application of the AP technique will only give rise to co-expressed gene clusters. In order to infer biologically plausible gene modules, we need a clustering technique to incorporate motif information and expression data to extract regulated gene expression patterns. In this paper, we propose to modify the AP clustering technique to find co-regulated gene expression patterns. In particular, we will define a

new similarity measure for AP to find a group of genes that not only is of similar expression pattern but also shares some same binding sites.

From motifs' position-weighted matrices (PWMs), we start with a matrix consisting of weights of n TFBSs in N genes. When the dataset is represented by a set of n of TFBSs, $T = \{t_1, t_2, \cdots, t_n\}$ and a set of N genes, $G = \{g_1, g_2, \cdots, g_N\}$ it can be viewed as a gene-TFBS matrix, $GT = [w(t_r, g_j)]$. Here $w(t_r, g_j)$ denotes the weight of the r-th TFBS in the j-th gene. Each weight indicates the measure that the r-th TFBS affects j-th gene. Given that the weight of TFBS r with g_{i} is $w(t_r, g_i)$ and the weight of TFBS r with g_{k} is $w(t_r, g_k)$, our assumption is that these two events are independent, so the pairwise joint weight of TFBS r with g_{i} and g_{k} is $w(t_r, g_i) * w(t_r, g_k)$. The total pairwise joint weight of TFBS r with g_{i} and g_{k} can be expressed as:

$$W = \sum_{r=1}^{n} w(t_r, g_i) * w(t_r, g_k). \tag{4}$$

In the algorithm of AP, we modify the pairwise similarity measure between g_{i} and g_{k} to be:

$$s(i,k) = -\sum_{i,k} (g_i - g_k)^2 + \lambda \sum_{r=1}^{n} w(t_r, g_i) * w(t_r, g_k), \tag{5}$$

where λ is a trade-off parameter (note that we set $\lambda = 0.05$ in our experiments). The first term in (5) is to help find a group of genes with similar expression pattern while the second term to enforce them sharing some same binding sites. Ideally, the clustering result will give us some clues about that this group of genes may indeed be regulated by one of the common TFBSs and therefore shows the same pattern as the corresponding TFBS activity level (i.e., corresponding to one particular columns of A).

2.2 Inferring Regulation Strength Matrix by Sparse Component Analysis (SCA)

In this section, we will employ the SCA approach to estimate the regulation strength matrix S, describing the relationships between TFBSs and gene populations. Here, we describe a regulation strength recovery algorithm based on Iterative Detection-Estimation [12]. An outline of the algorithm is described as follows:

Regulation Strength Recovery Algorithm

Loop

1. Detection Step: Starting with a previous estimate (or the initialization) of the regulation strength vectors $\boldsymbol{s}$, roughly detect which TFBSs are "active";

2. Estimation Step: Having the indices of "active" TFBSs, obtain a new estimate of $\boldsymbol{s}$ by finding a solution of $\boldsymbol{x} = A\boldsymbol{s}$ whose active indices better coincide with those predicted by the detection step.

Until converged

The term "active" is used to refer to the TFBSs having "considerably large" strengths. In the detection step, let π_0 be the probability of s_i being inactive ($\pi_0 \simeq 1$ to insure sparsity). Then we use Gaussians to model the values of an inactive TFBS and an active TFBS respectively by $N(0,\sigma_0^2), N(0,\sigma_1^2)$, where $\sigma_0^2 \ll \sigma_1^2$. We may formulate the problem in terms of a binary hypothesis testing. The vector $\boldsymbol{x} = s_1\boldsymbol{a}_1 + \sum_{i=2}^{n} s_i\boldsymbol{a}_i$ is observed, and we need to detect which of the following two hypotheses have occurred;

$$\begin{aligned} H_0 &: s_1 \sim N(0,\sigma_0^2) \\ H_1 &: s_1 \sim N(0,\sigma_1^2) \end{aligned}. \tag{6}$$

Defining $\mu \triangleq \sum_{i=2}^{n} s_i\boldsymbol{a}_1^T\boldsymbol{a}_i$, we will have $t = \boldsymbol{a}_1^T\boldsymbol{x} = s_1 + \mu$. The equivalent test in terms of the sufficient statistics, *t*, may be stated as $H_i : t \sim N(\mu,\sigma_i^2)$ for $i = 0,1$.

However, it appears that implementing the optimal test for activity of s_1 requires the knowledge of all the other TFBSs. Since they are also unknown parameters, we have to replace them with their estimates. Therefore, the resulting sub-optimal test is obtained as follows:

$$\left| \boldsymbol{a}_1^T\boldsymbol{x} - \sum_{i=2}^{n} \hat{s}_i\boldsymbol{a}_1^T\boldsymbol{a}_i \right| > \varepsilon . \tag{7}$$

This test is conducted for *n* TFBSs. After determining the activity status, we try to determine the actual values of the TFBSs. For the sake of the discussion, assume that the first k TFBSs, $\{s_i\}_{i=1}^{k}$, have been detected to be inactive. Then the approximation of the regulation strength vectors can be obtained by the following optimization problem [7]:

$$\min \sum_{i=1}^{k} s_i^2 \quad \text{subject to} \quad \boldsymbol{x} = A\boldsymbol{s} . \tag{8}$$

3 Experimental Results

In this section, we will report the experimental results on an estrogen dependent profiling study of breast cancer. In particular, we will describe the data set used to test our two-step mSD approach, the extracted motifs related to estrogen receptor (ER) and their PWMs obtained, and the transcription modules recovered by our method.

3.1 Dataset Description

Estrogen has a profound impact on human physiology and affects numerous genes. The classical estrogen reaction is mediated by its receptors (ERs), which bind to the estrogen response elements (EREs) in target gene's promoter region. In [13], the authors utilized an integrated genome-wide molecular and computational approach to characterize the interaction between the activated estrogen receptor and the regulatory elements of candidate target genes.

The authors treated the estrogen-dependent T-47D ER+ breast cancer cell line with 17β-estradiol (E2) and with E2 in combination with either the pure anti-estrogen ICI

182,780 (ICI) or the protein synthesis inhibitor CHX and performed high-resolution time-course gene-expression analyses using spotted oligonucleotide (60-mers) microarrays containing probes representing around 19,000 human genes. Samples were harvested on an hourly basis for the first 8 hours (0-8 hours) following hormone treatment and bi-hourly for the next 16 hours (10-24 hours) for a total of 16 time points surveyed [13]. We will compare the results from the 16 time points E2-treated samples with 16 time points E2+ICI-treated samples.

The overall objective of this analysis is to search for the relationships of putative or potential target genes and the likely TFBSs that are functionally associated with transcriptional regulatory networks regulated by ER. We collected not only genes that identified by [13] as putative ER target genes, but also some other gene sets. For example, in [14], a subclass of estrogen response genes that are computed by matching ERE frames of a test set of 60 known estrogen responsive genes to the collection of over 18,000 human promoters. We also included gene sets that were collected by pathway database (Biocarta, KEGG, etc) and related to breast cancer signaling pathway. Finally, we formed a time course data set consisting 1288 human genes in two treatment conditions (E2 and E2+ICI).

3.2 Motif Analysis for Binding Information

From TRANSFAC database and ChIP-on-chip experiments [15], 44 breast cancer and ER-related transcription factors were selected for motif analysis (shown in Table 1). First, the upstream regions of the genes can be extracted from the database PromoSer [16]. Secondly, Match™ [17] can be used to search the transcription factor binding site (TFBS) in each upstream region, which outputs the core similarity and matrix similarity for each matched motif. Third, Match™ searches the TFBS by its position-weighted matrices (PWMs), which can be extracted from TRANSFAC 11.1 Professional database [8]. In the last step, according to the PWMs, a motif-score will be calculated for each TF-gene pair, which can be regarded as connection strength information.

Note that motif is a relative short sequence pattern, thus the topology from motif information is just a rough estimation, possibly with many false positives and false negatives. Although the motif information is not reliable according to one specific gene with one specific transcription factor, we can still infer some key transcription factor activities from the whole genome population, with the initialization of regulation strength *S* in our method by PWM.

Table 1. 44 breast cancer and ER-related transcription factors from TRANSFAC database

V$OCT1_03	V$ER_Q6	V$BRCA_01	V$STAT3_01	V$HOX13_01
V$HMGIY_Q3	V$VDR_Q3	V$HMGIY_Q6	V$OCT1_01	V$HNF1_Q6
V$OCT1_02	V$FOXO1_02	V$FOXO1_01	V$OCT1_04	V$OCT1_Q6
V$USF2_Q6	V$NF1_Q6_01	V$STAT1_03	V$VDR_Q6	V$DR3_Q4
V$ER_Q6_02	V$STAT1_02	V$ETS_Q4	V$ETS_Q6	V$STAT3_02
V$AP2GAMMA_01	V$OCT1_Q5_01	V$OCT1_B	V$OCT1_06	V$P53_01
V$OCT1_07	V$OCT1_05	V$PPARG_01	V$P53_02	V$STAT1_01
V$PR_02	V$ZBRK1_01	V$PPARG_02	V$HNF1_01	V$HNF3ALPHA_Q6
V$ETS1_B	V$ERR1_Q2	V$HNF1_C	V$PR_01	

3.3 Results on Estrogen Receptor-Related Transcriptional Modules

We have applied our two-step mSD approach to the E2/E2+ICI data set to identify active ER-related TFBSs and their downstream target genes. In the first step, we applied motif information guided AP clustering technique to estimate the TFBS activity matrix *A*. In the second step, we applied the SCA approach to iteratively estimate the regulation strength matrix *S* that defines the transcriptional modules.

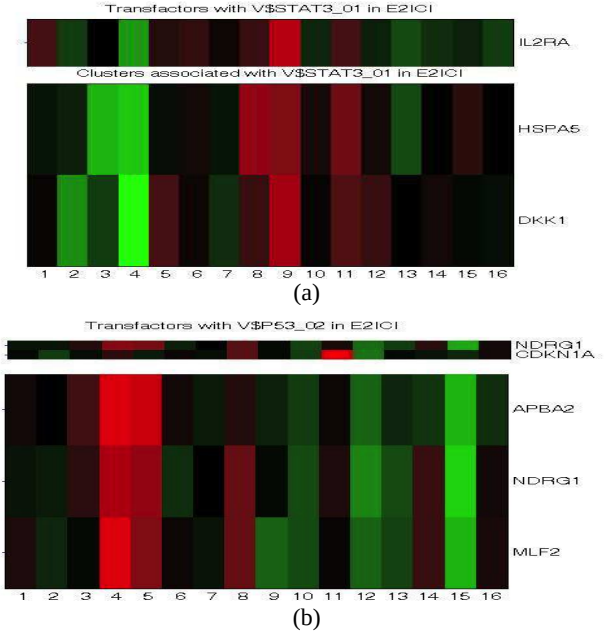
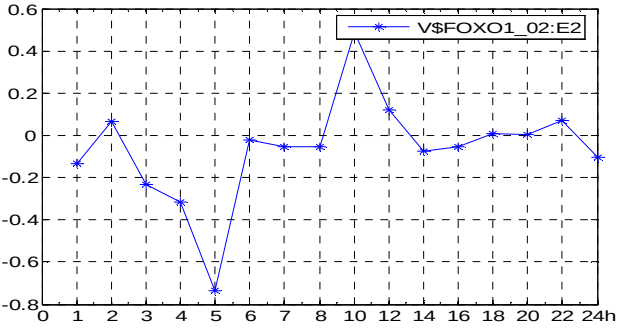


Fig. 1. Heatmaps of the TFs and their putative targets. (a) the TF (IL2RA) searched from TRANSFAC which has the binding site V$STAT3_01 and the cluster which has been mapped to be the predictor of the profile of TFBS V$STAT3_01; (b) TFs (NDGR1; CDKN1A) searched from TRANSFAC which have the binding site V$P53_02 and the cluster which has been mapped to be the predictor of the profile of TFBS V$P53_02.

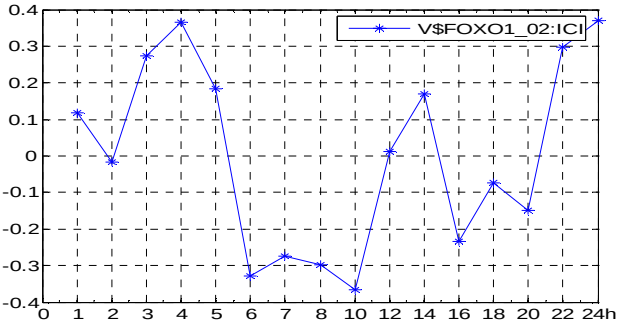
After clustering, we map the TFBSs to the inferred clusters based on previous knowledge about their activity profiles. To infer the columns of *A* (TFBS's activity), we assume that the activity levels of transcription factors are proportional to their mRNA levels. Thus, from PWMs, we extracted the gene set which has the high weights to be regulated by one specific TFBS, and we checked the intersection with every cluster. If one of the clusters and the gene set from PWMs have the largest overlap, we mapped the "centroid" of the cluster as the corresponding TFBS's activity. The results are shown in Fig. 1. In Fig. 1(a), the upper panel presents the expression profile of a TF (extracted from TRANSFAC database) that has the particular binding site (V$STAT3_01 in this case). And the lower panel of the figure shows the representative expression profiles of the cluster which has been mapped to mimic V$STAT3_01's activity level. From the heatmaps, we can see that the patterns of the

predicators (shown in the lower part of the figure) and the predicted one (shown in the upper part of the figure) are very similar. Another example is also given in Fig. 1(b) with the binding site V$P53_02. These results confirm that the expression levels of target genes are indeed good predictors of the corresponding TFBSs when we incorporate prior motif information into our modified AP clustering method.

After the TFBS activity matrix *A* has been recovered in the first step, the SCA approach is followed to iteratively estimate the active TFBSs and their regulation strength matrix *S*. In this study, we have detected about 20 active TFBSs in both E2 and E2+ICI conditions on T-47D ER+ breast cancer cell line. Comparing the transcriptional modules defined by the estimated *S* in two conditions, we can divide them into two categories: (1) condition-independent transcriptional modules and (2) condition-specific transcriptional modules. Below, we focus on one of the condition-specific modules recovered by our mSD approach for a detailed discussion.



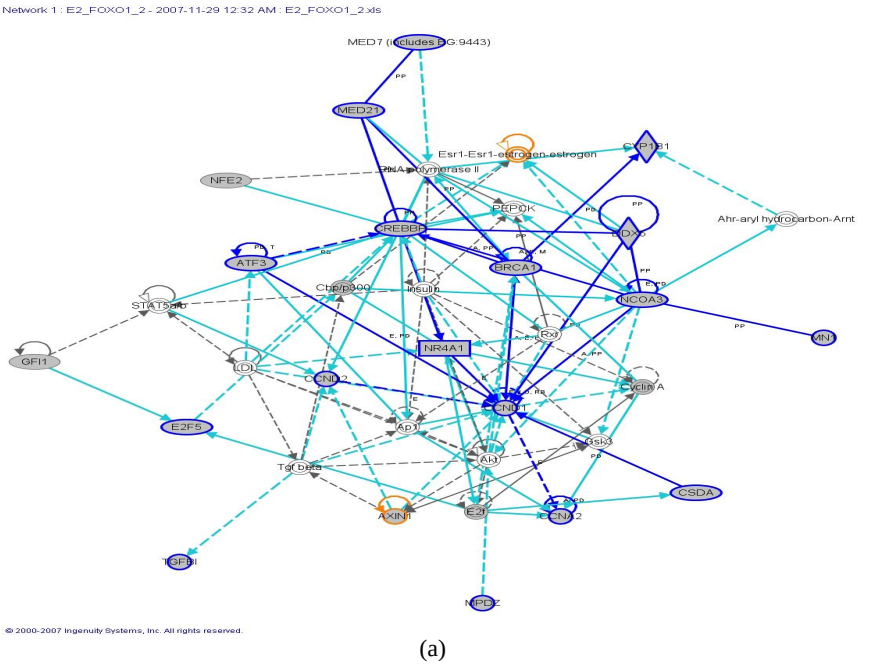
(a)



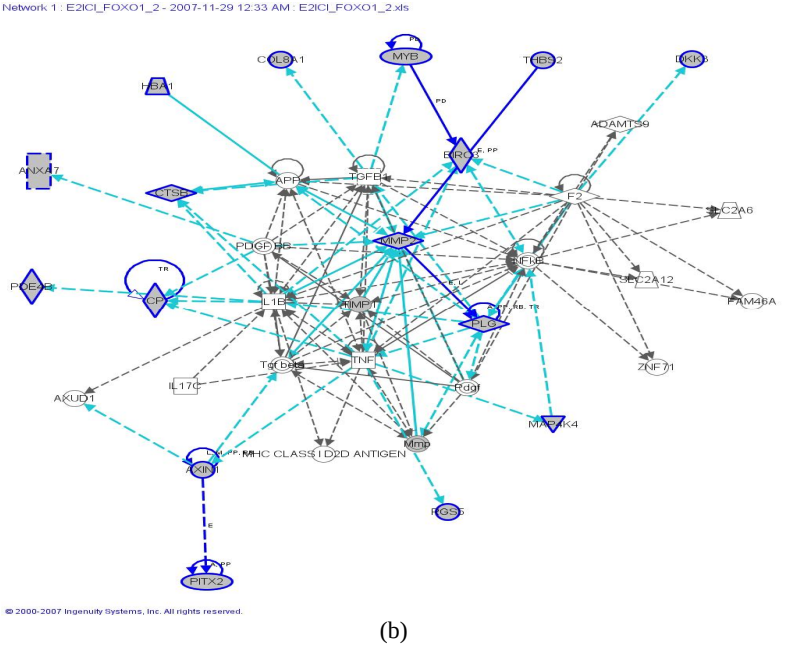
(b)

Fig. 2. TFBS V$FOXO1_02's activity levels in (a) E2 and (b) E2+ICI, respectively

We observed that one of the active TFBSs, $FOXO1_02, is of interesting TFBS activities in two conditions. Its patterns in two different conditions are nearly complementary, as shown in Fig. 2. As we know, genes with promoter regions [-2kb, 2kb] around transcription start site containing the motif GNNTTGTTTACNTT, which matches annotation for FOXO1A: forkhead box O1A (rhabdomyosarcoma). It has been reported [15] that the primary interaction of estrogen receptor with chromatin



(a)



(b)

Fig. 3. Two networks of target genes of TFBS V$FOXO1_02 in two different conditions: E2/E2+ICI

can occur either through direct interaction with an ERE or through a tethering mechanism involving AP-1 factors with C/EBP, Oct and *Forkhead* motifs functioning as adjacent binding sites for cooperating factors. Furthermore, we extracted the target genes of V$FOXO1_02 inferred by our mSD method and fed them into Ingenuity Pathway Analysis (IPA; http://www.ingenuity.com/). Fig. 3 illustrates the two gene networks in which those target genes are involved in two different conditions. It can be clearly seen that in the E2 condition, the top associated network functions are: cell cycle and gene expression, in which ESR1-estrogen complex is involved. However, in the E2+ICI condition, the top associated network function has change to cellular movement, in which ESR1-estrogen complex is not involved. This is consistent to that ICI is an anti-estrogen drug that blocks the involvement of ESR1-estrogen complex [13].

4 Conclusions

In this paper, we have developed a new approach, namely motif-guided sparse decomposition (mSD), for transcriptional module identification. The mSD approach combines the motif information and gene expression data with an emphasis on the interplay of co-expression and co-regulation. Motif information is first used to guide a clustering technique to find regulated gene expression patterns; sparse component analysis is then used to identify active transcription factor binding sites (TFBSs) and their regulation strengths on the target genes. Experimental results on an estrogen receptor profiling study have demonstrated that the mSD approach can help identify condition-specific transcriptional modules that showing distinct TFBS activities and target genes in different conditions.

Acknowledgments. This research was supported in part by NIH Grants (NS29525-13A, EB000830, CA109872 and CA096483) and a DoD/CDMRP grant (BC030280).

References

1. Speed, T.: Statistical Analysis of Gene Expression Microarray Data. Chapman & Hall/CRC (2003)
2. Nguyen, D.H., D'Haeseleer, P.: Deciphering principles of transcription regulation in eukaryotic genomes. Mol. Syst. Biol. 2 (2006)
3. Roth, F.P., et al.: Finding DNA regulatory motifs within unaligned noncoding sequences clustered by whole-genome mRNA quantitation. Nat Biotech 16(10), 939–945 (1998)
4. Jin, V.X., et al.: A computational genomics approach to identify cis-regulatory modules from chromatin immunoprecipitation microarray data–A case study using E2F1. Genome Res. 16(12), 1585–1595 (2006)
5. Smith, A.D., et al.: Mining ChIP-chip data for transcription factor and cofactor binding sites. Bioinformatics 21(suppl. 1), 403–412 (2005)
6. Liao, J.C., et al.: Network component analysis: Reconstruction of regulatory signals in biological systems. Proceedings of the National Academy of Sciences 100(26), 15522–15527 (2003)

7. Georgiev, P., Theis, F., Cichocki, A.: Sparse component analysis and blind source separation of underdetermined mixtures. Neural Networks, IEEE Transactions 16(4), 992–996 (2005)
8. Schacherer, F., et al.: TRANSFAC: an integrated system for gene expression regulation. Nucleic Acids Research 28, 316–319 (2000)
9. Hartigan, J.A., Wong, M.A.: A K-means clustering algorithm. App. Statist. 28, 100–108 (1978)
10. Kohonen, T.: Self-Organizing Maps. Springer, NY (1997)
11. Frey, B.J., Dueck, D.: Clustering by Passing Messages Between Data Points. Science 315(5814), 972–976 (2007)
12. Arash Ali, A., Massoud, B.-Z., Christian, J.: A Fast Method for Sparse Component Analysis Based on Iterative Detection-Estimation. In: AIP Conference Proceedings, vol. 872(1), pp. 123–130 (2006)
13. Lin, C.-Y., et al.: Discovery of estrogen receptor alpha target genes and response elements in breast tumor cells. Genome Biology 5(9), R66 (2004)
14. Tang, S., et al.: Computational method for discovery of estrogen responsive genes. Nucl. Acids Res. 32(21), 6212–6217 (2004)
15. Carroll, J.S., et al.: Genome-wide analysis of estrogen receptor binding sites. Nat. Genet. 38(11), 1289–1297 (2006)
16. Halees, A.S., Leyfer, D., Weng, Z.: PromoSer: A large-scale mammalian promoter and transcription start site identification service. Nucleic Acids Res. 31(13), 3554–3559 (2003)
17. Kel, A.E., et al.: MATCH: A tool for searching transcription factor binding sites in DNA sequences. Nucleic Acids Res. 31(13), 3576–3579 (2003)

A Novel Metric for Redundant Gene Elimination Based on Discriminative Contribution

Xue-Qiang Zeng[1,2], Guo-Zheng Li[1,2,⋆], Jack Y. Yang[3], and Mary Qu Yang[4]

[1] Institute of System Biology, Shanghai University, Shanghai 200444, China
[2] School of Computer Engineering and Science, Shanghai University, Shanghai 200072, China
gzli@shu.edu.cn
[3] Harvard Medical School, Harvard University, Cambridge, Massachusetts 02140-0888 USA
[4] National Human Genome Research Institute National Institutes of Health (NIH) U.S., Department of Health and Human Services Bethesda, MD 20852 USA

Abstract. As a high dimensional problem, analysis of microarray data sets is a hard task, where many weakly relevant but redundant features hurt generalization performance of classifiers. There are previous works to handle this problem by using linear or nonlinear filters, but these filters do not consider discriminative contribution of each feature by utilizing the label information. Here we propose a novel metric based on discriminative contribution to perform redundant feature elimination. By the new metric, complementary features are likely to be reserved, which is beneficial for the final classification. Experimental results on several microarray data sets show our proposed metric for redundant feature elimination based on discriminative contribution is better than the previous state-of-arts linear or nonlinear metrics on the problem of analysis of microarray data sets.

1 Introduction

The rapid advances in gene expression microarray technology enable simultaneously measuring the expression levels for thousands or tens of thousands of genes in a single experiment [1]. Analysis of microarray data presents unprecedented opportunities and challenges for data mining in areas such as gene clustering, class discovery, and sample classification [2,3,4]. In sample classification, a microarray data set is provided as a training set of labeled samples. The task is to build a classifier that accurately predicts the classes of novel unlabeled samples. A typical data set has thousands of genes but only a small number of samples (often less than a hundred). The number of samples is likely to remain small at least for the near future due to the expense of collecting microarray samples [5]. The nature of relatively high dimensionality but small sample size in microarray data cause the known problem of "curse of dimensionality". Therefore, selecting a small number of discriminative genes from thousands of genes is essential for successful sample classification.

⋆ Corresponding author.

I. Măndoiu, R. Sunderraman, and A. Zelikovsky (Eds.): ISBRA 2008, LNBI 4983, pp. 256–267, 2008.

Feature selection, a process of choosing a subset of features from the original ones, is frequently used as a preprocessing technique in data mining. It has been proved effective in reducing dimensionality, improving mining efficiency, increasing mining accuracy, and enhancing result comprehensibility [6]. In the field of bioinformatics, the most commonly used procedures of feature selection (gene selection) are based on a score which is calculated for all genes individually and genes with the best scores are selected [7]. Feature selection procedures output a list of relevant genes which may be experimentally analyzed by biologists. This method is often denoted as univariate feature selection (filter methods), whose advantages are its simplicity and interpretability.

As to analysis of microarray data sets, whose speciality is the huge amount of genes with few samples, it is believed that there are many weakly relevant but redundant genes among thousands of genes. Preserving the most discriminative genes and reducing other irrelevant and redundant ones is the target of feature selection. However, because the interactions and correlations among genes are omitted, filter methods fail to remove redundant genes. The scores they assign to correlated genes are too similar, and none of the genes are strongly preferred over others. Redundancy among selected genes results in two problems. One problem is that redundant features in the selected subset reduce the comprehensive representation of target labels. The other one is that redundant genes increase the dimensionality of the selected gene set, which affect the mining performance on the small sample [5].

The issue of redundancy among genes is recently raised in the literatures of gene selection [8,9]. Researchers have proposed several methods to reduce the redundancy among genes. Ding and Peng proposed the minimum Redundancy-Maximum Relevance (mRMR) method [8,10], which requires that selected discriminative features are maximally dissimilar to each other. Ding and Peng define the feature redundancy by the metric of mutual information. Yu and Liu proposed the Fast Correlation-Based Filter (FCBF) method [11,12,9], which is based on approximate Markov blanket. FCBF eliminates redundant features by iteratively selecting predominant features from relevant ones where the feature redundancy is measured by symmetrical uncertainty.

However, the feature redundancy metrics used by these methods can not estimate the redundancy properly, because without consideration of the label information, the pair-wise redundancy scores solely calculated by the given two features do not faithfully reflect the discriminative ability similarity between them. For example, two highly correlated features, whose differences are minor but happen to be different critical discriminative ability, maybe be considered as a pair of redundant features by normal metrics. Hence, reducing any one of them will decrease classification accuracy. The fact is that the existing methods only directly compare the similarity of the numerical values between two features, but not compare the similarity of discriminative ability between two features, i.e. the distribution of correctly predicted examples by using two single features.

In order to solve this problem, we propose a novel metric of redundancy based on DIScriminative Contribution (DISC), which directly compares the classification

distribution between two features. By measuring the discriminated examples on each single feature, DISC defines redundant genes which have few discriminative ability contribution to a given one. We also compare our metric with state-of-arts feature redundancy metrics, including linear [13] and non-linear ones [12]. Experiments on several real world microarray data sets demonstrate the outstanding performance of our metric.

This paper is organized as follows. Section 2 discusses the metrics of discriminative ability and usual metrics of redundancy. In section 3, a novel metric is presented in detail. Then, data sets, experiment settings and evaluation methods are described in section 4. We show the results and discussions in section 5. Finally, conclusions are given in section 6.

2 The Previous Metrics

Discriminative ability (predictive ability) is a general notion which can be measured in various ways and be used to select significant features for classification. Many effective metrics had been proposed such as t-statistic, information gain, χ^2 statistic, odds ratio *etc.* [14,15]. These metrics have also often been used as indicative scores in filter feature selection methods to sort features, and then some top ranked features are retained which are supposed to be the essential ones for classification.

However, most of these discriminative metrics only give a discriminative score to each individual feature, which can not be used to compare the similarity between two features. Because most discriminative information are missed when only one numerical score is retained for each feature. So, when feature ranking is not the only application, discriminative metric should preserve much more information than before.

For the task of feature selection, we want to eliminate the redundant features and only retain the complement ones. But there exist many redundant features in the top ranked feature set produced by the filter methods. The redundant features increase the dimensionality and contribute little for the final classification. In order to eliminate redundant features, some powerful discriminative metrics which can be used to measure the feauture redundancy directly are needed.

In normal cases, notions of feature redundancy are in terms of feature correlation. It is widely accepted that two features are redundant to each other if their values are completely correlated. But in fact, it may not be so straightforward to determine feature redundancy when a feature is correlated with a set of features. The widely used way is approximate the redundancy of feature set by only considering the pair-wise feature redundancy, *i.e.* Yu and Liu [11] used the pair-wise symmetrical uncertainty to measure the feature redundancy.

2.1 Linear Correlation Metrics

For linear cases, the most well known pair-wise redundancy metric is the linear correlation coefficient. Given a pair of features (x, y), the definition of the linear

correlation coefficient $\mathrm{cor}(x, y)$ is:

$$\mathrm{cor}(x, y) = \frac{\sum_i (x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\sum_i (x_i - \overline{x})^2} \sqrt{\sum_i (y_i - \overline{y})^2}} \tag{1}$$

where $\overline{x}$ and $\overline{y}$ are the mean of x and y respectively. The value of $\mathrm{cor}(x, y)$ lies between -1 and 1. If x and y are completely correlated, $\mathrm{cor}(x, y)$ takes the value of 1 or -1; if x and y are independent, $\mathrm{cor}(x, y)$ is zero. It is a symmetrical metric.

The linear correlation coefficient has the advantage of its efficiency and simplicity, but it is not suitable for redundant feature elimination when classification is the final target, since it does not use any label information. For example, two highly correlated features, whose differences are minor but happen to causing different critical discriminative ability, may be considered as a pair of redundancy features. Reducing any one of them will decrease the classification accuracy. Guyon has also pointed out that very high variable correlation (or anti-correlation) does not mean absence of variable complementarity [14]. The problem of the linear correlation coefficient is that it measures the similarity of the numerical values between two features, but not that of discriminative ability.

2.2 Non-Linear Correlation Metrics

Many non-linear correlation metrics are based on the information-theoretical concept of entropy, a metric of the uncertainty of random variables. Entropy of a variable x is defined as:

$$H(x) = -\sum_i p(x_i) \log_2 P(x_i) \tag{2}$$

and the entropy of x after observing values of another variable y is defined as:

$$H(x|y) = -\sum_j P(y_j) \sum_i p(x_i|y_j) \log_2 P(x_i|y_j) \tag{3}$$

where $P(x_i)$ is the prior probabilities for all values of x, and $P(x_i|y_j)$ is the posterior probabilities of x given the values of y. The amount by which the entropy of x decreases reflects additional information about x provided by y and is called mutual information [16], given by

$$I(x|y) = H(x) - H(x|y) \tag{4}$$

According to this metric, feature y is regarded more correlated to feature x than to feature z, if $I(x|y) > I(z|y)$. It is easy to prove that mutual information is a symmetrical metric.

Since mutual information tends to favor features with higher values, it should be normalized with their corresponding entropy. One of the most widely used normalized mutual information is Symmetrical Uncertainty (SU) ([17]), which is defined as:

$$\mathrm{SU}(x|y) = 2\left[\frac{I(x|y)}{H(x) + H(y)}\right] \tag{5}$$

SU compensates for mutual information bias toward features with higher values and restricts its values to the range [0,1]. A value of 1 indicates that knowing the values of either feature completely predicts the values of the other; a value of 0 indicates that x and y are independent. In addition, it also treats a pair of features symmetrically. Entropy-based metrics only handle nominal or discrete features, and therefore continuous features need to be discretized beforehand.

Normal entropy-based metrics also do not take the label information into consideration either. Hence, the problem of non-linear methods is similar to that of linear ones. Previously proposed redundancy metrics estimate the similarity between two features only by the numerical values. But what is more important, we do not know whether the dissimilarly parts between them are essential for the final classification.

To overcome this problem, the discriminative contribution of the dissimilar parts should be examined in a measurable way. In other words, the discriminative abilities of features should be recorded in a comparable manner. Then, the similarity of discriminative abilities between features can be used as a good estimation of feature redundancy.

3 The Proposed Novel Metric

Since tumor classification by using microarray data sets is a supervised problem, DIScriminative Contribution (DISC) of each genes is critical to performance of classifiers. If we perform redundant gene elimination, we should remove the features with little DISC. Here we define DISC based on training accuracy of single features , i.e. one classifier is built on each feature. From the classifier, we can precisely record which examples are correctly distinguished by the given feature. Both linear and non-linear classifiers can be used, here in simplification, we only consider a linear binary classifier here.

The classification function on feature x is defined as:

$$\hat{y} = \mathrm{sgn}((\overline{x}_+ - \overline{x}_-)(x - \frac{n_+\overline{x}_+ + n_-\overline{x}_-}{n_+ + n_-})) \tag{6}$$

where $\hat{y}$ is the predicted label, $\overline{x}_+$ and n_+ are the mean value (centroid) and the number of positive samples, $\overline{x}_-$ and n_- have the similar means of negative samples. A new example is predicted as the class whose centroid is closer to the given example. The computation complexity of this linear classifier is $O(n)$, $(n = n_+ + n_-)$.

Similar with normal discriminative metrics, training accuracy by testing on the training examples is used to represent the discriminative ability. High training accuracy means the corresponding feature has great discriminative ability. In most cases, only a part of the training samples can be correctly separated by this simple classifier.

But dissimilar with normal discriminative metrics, we have a predicted vector for each feature, which is supposed to be the approximation of classification distribution. Furthermore, the discriminative contribution can be estimated by comparing the corresponding classification distribution.

Table 1. Discriminative Cross Table

C_1 \ C_2	true	false
true	a	b
false	c	d

Given two features x_1 and x_2, two classifiers C_1 and C_2 are constructed. Feeding the whole training set back to the classifiers, the differences of the correctly classified samples of C_1 and C_2 are recorded in Table 1.

In Table 1, $a+b+c+d$ equals to the size of the training set n. The values of $\frac{a+b}{n}$ and $\frac{a+c}{n}$ are training accuracy of C_1 and C_2 respectively. The score of $a+d$ measures the similarity of the features, and the score of $b+c$ measures the dissimilarity. When both b and c equals with zero, the two features x_1 and x_2 are considered having exactly the same discriminative abilities.

In order to eliminate redundant features, we examine whether the contribution of the additional feature is significant to the given one. The additional feature is considered as redundant one only when its contribution is tiny. Based on the pair-wise discriminative contribution, we propose a novel metric of redundancy based on discriminative Contribution (DISC). The DISC value of x_1 to x_2, which represents the x_2's redundancy to x_1, is defined as follows:

$$\mathrm{DISC}(x_1, x_2) = \frac{1}{2} \times \left(\frac{a}{a+c} + \frac{d}{b+d}\right) \tag{7}$$

where $a+c$ is the samples which can be discriminated by C_2, within it, a is the samples which can also be discriminated by C_1. So the proportion of $\frac{a}{a+c}$ measures how much discriminative abilities of C_2 are covered by C_1. $b+d$ is the samples which could not be discriminated by C_2, within it, d is the samples which could not be rightly classified even by the collaboration of C_1 and C_2. So the proportion of $\frac{d}{b+d}$ represents the useless extents of C_2 which are same with that of C_1.

Based on the idea of discriminative contribution, DISC measures the x_2's redundancy to x_1 , which gives the same weights to the discriminative ability $\frac{a}{a+c}$ and the discriminative useless extent $\frac{d}{b+d}$.

The DISC score varies from 0 to 1, and takes 1 only when both b and c are 0. In this case, we consider x_2 is completely redundant to x_1. On the other hand, when the DISC value is 0, both a and d are 0, we suppose the discriminative ability of x_2 is complementary to that of x_1.

The pair-wise DISC metric is asymmetrical, so it is not suitable to be used as a distance metric. The computation complexity of DISC between two single features is $O(n)$, which is same with the normal linear correlation coefficient. Furthermore, the classifiers are needed to be built only once in the whole feature selection algorithms.

DISC is proposed in a linear way, which shows in two respects, one is the linear classifier, another is the linear way of counting the cross discriminative abilities. The microarray problems meet the assumption, since most microarray

data sets are binary classification problems, where each gene has equal position to perform classification. As for the non-linear cases, it is much more complicated. Because directly comparing predicted vectors is maybe not suitable for non-linear classifiers. We will examine the non-linear problem in future works.

4 Experiments

4.1 Data Sets

Four microarray data sets used in our study are listed in Table 2. They are briefly described as below, and the corresponding C4.5 format versions are available at [18].

Breast Cancer. Van't Veer *et al.* used DNA microarray analysis on primary breast tumours and applied supervised classification to identify the significant genes for the disease [19]. The data contains 97 patient samples, 46 of which are from patients who had developed distance metastases within 5 years (labeled as "relapse"), the rest 51 samples are from patients who remained healthy from the disease after their initial diagnosis for interval of at least 5 years (labeled as "non-relapse"). The number of genes is 24,481 and the missing values of "NaN" are replaced with 100.

Colon. Alon *et al.* used Affymetrix oligonucleotide arrays to monitor expressions of over 6,500 human genes with samples of 40 tumor and 22 normal colon tissues [3]. Expression of the 2,000 genes with the highest minimal intensity across the 62 tissues were used in the analysis.

DLBCL. [20] used gene expression data to analysis distinct types of diffuse large B-cell lymphoma (DLBCL). DLBCL is the most common subtype of non-Hodgkin's lymphoma. There are 47 samples, 24 of them are from "germinal centre B-like" group and 23 are "activated B-like" group. Each sample is described by 4,026 genes. The missing values in the data set are replaced by the corresponding averaged column values.

Leukemia. [2] consists of 72 bone marrow samples with 47 ALL and 25 AML. The gene expression intensities are obtained from Affymetrix high-density oligonucleotide microarrays containing probes for 7,129 genes.

Table 2. Experimental Data Sets

Data Sets	Samples	Class Ratio	Features
Breast Cancer	97	46/51	24,481
Colon	62	22/40	2,000
DLBCL	47	23/24	4,026
Leukemia	72	25/47	7,129

4.2 Experimental Settings

We use the stratified 10-fold cross-validation procedure, where each data set was split into ten subsets of equal size. Each subset is used as a test set once, and the corresponding left subsets are merged together and used as the training set. Within each cross-validation fold, the data are standardized. The expressions of the training set are transformed to zero mean with unit standard deviation across examples, and the test set are transformed according to the means and standard deviations of the corresponding training set.

The classifiers, i.e. linear Support Vector Machine (SVM) with $c = 1$, Naïve Bayes (NB) and k Nearest Neighbor (kNN) with $k = 1$ are used, which are trained on the training set to predict the label of the test examples. The cross-validation procedure is repeated ten times, and the mean classification accuracy (ACC)

$$\text{ACC} = \frac{\text{number of correctly pedicted examples}}{\text{total number of test examples}}$$

is used to measure the performance.

5 Results and Discussions

5.1 Results

In order to examine performance of different redundancy metrics, the famous redundant feature selection framework of FCBF [11,12] is used in our experiments. Under the unified framework, we compare our proposed redundancy metric with LInear Correlation (LIC) and Symmetrical Uncertainty (SU). Finally, three widely used classifiers i.e. SVM, NB and kNN are applied to examine the performance. The comparative results are showed in Tables 3∼5, where ACC±std are the statistical mean values with its standard deviation, since the cross-validation procedure is performed ten times.

From Tables 3∼5, we can see that:

(1) DISC is the best one among the three redundancy metrics in average. Of four data sets, DISC is always the best one on two data sets, only in three

Table 3. Comparative results of different redundancy metrics by using SVM

	LIC		SU		DISC		Full set	
Data Sets	ACC±std	Dim.	ACC±std	Dim.	ACC±std	Dim.	ACC±std	Dim.
BreastCancer	0.5267±0.00	293.00	0.6671±0.03	97.79	0.6880±0.03	166.99	0.6784±0.01	24,481
Colon	0.6493±0.01	3.60	0.8333±0.02	14.04	0.8560±0.01	30.26	0.8493±0.02	2,000
DLBCL	0.8930±0.03	18.59	0.9170±0.03	48.92	0.9480±0.02	212.70	0.9325±0.03	4,026
Leukemia	0.8648±0.05	28.35	0.9461±0.01	45.34	0.9621±0.01	256.90	0.9791±0.01	7,129
Average	0.7334±0.02	85.89	0.8409±0.02	51.52	0.8635±0.02	166.71	0.8598±0.02	9,409

Table 4. Comparative results of different redundancy metrics by using Naïve Bayes

	LIC		SU		DISC		Full set	
Data Sets	ACC±std	Dim.	ACC±std	Dim.	ACC±std	Dim.	ACC±std	Dim.
BreastCancer	0.5267±0.00	293.00	0.5363±0.02	97.79	0.6703±0.02	166.99	0.5551±0.01	24,481
Colon	0.6912±0.06	3.60	0.7990±0.03	14.04	0.7702±0.03	30.26	0.5745±0.02	2,000
DLBCL	0.9585±0.03	18.59	0.9535±0.02	48.92	0.9660±0.02	212.70	0.9425±0.03	4,026
Leukemia	0.9198±0.02	28.35	0.9593±0.01	45.34	0.9705±0.01	256.90	0.9789±0.01	7,129
Average	0.7740±0.03	85.89	0.8120±0.02	51.52	0.8443±0.02	166.71	0.7628±0.02	9,409

Table 5. Comparative results of different redundancy metrics by using kNN

	LIC		SU		DISC		Full set	
Data Sets	ACC±std	Dim.	ACC±std	Dim.	ACC±std	Dim.	ACC±std	Dim.
BreastCancer	0.4787±0.02	293.00	0.6471±0.04	97.79	0.6871±0.04	166.99	0.5932±0.02	24,481
Colon	0.6757±0.06	3.60	0.7652±0.04	14.04	0.7676±0.04	30.26	0.7529±0.02	2,000
DLBCL	0.8585±0.03	18.59	0.8840±0.04	48.92	0.9010±0.03	212.70	0.7575±0.03	4,026
Leukemia	0.8334±0.04	28.35	0.9532±0.03	45.34	0.9550±0.02	256.90	0.8725±0.01	7,129
Average	0.7116±0.04	85.89	0.8124±0.04	51.52	0.8277±0.03	166.71	0.7440±0.02	9,409

cases, DISC performs slightly worse than others, i.e. on the Leukemia data set DISC performs slightly worse than the full set by using SVM and NB, and on the Colon data set DISC performs slightly worse than SU does by using NB.

(2) Linear correlation (LIC) is the worst one among all the metrics in average. The results of LIC are only better than those without any feature selection in case of using NB and kNN on the Colon and DLBCL data sets.

(3) Results of symmetrical uncertainty (SU) are worse than those of full data sets without feature selection.

(4) As for the number of selected features, LIC obtains the least size on three data sets, it is in tens, and SU also obtains feature subsets in tens, while DISC selects one to two hundreds of features. Sizes of the whole data sets are in two to twenty-four hundreds, so all metrics succeed compress the features.

5.2 Discussions

Experimental results has shown DISC not only improves generalization performance of classifiers on original data sets but also greatly reduce the number of used genes. We have such considerations:

(1) The fact that performance of linear correlation coefficient is much worse than the other two metrics indicates that taking the concrete separable examples between different features into consideration is important to estimate the

pair-wise redundancy. For linear correlation, when the absolute value is not 1, we can not make out whether the different parts between two features are useless for classification of the test sample.

(2) Although the novel metric is proposed by using linear classifiers. its performance is better than that of non-linear metrics i.e. SU in our experiments. One reason is that DISC can find the complimentary features for the existing strong relevant features. Another possible reason is the characteristic of microarray data sets, whose features are rather more than the examples.

(3) From the Table3, it can also be seen that the ACC score of the full set is very close to the best one of DISC in average. This is because the SVM classifier is good at handling high dimensional problems and redundant features, where feature selection is hard to improve the performance of SVM. But DISC still improves generalization performance of SVM on three out of four data sets. As for the NB and kNN classifiers, high dimensionality obscures the procedures of classification and feature selection is rather helpful for them.

(4) From the results we know, if we want to analyze the genes, we may use the symmetrical uncertainty metric, which obtain similar prediction performance with less features, i.e. one percent of total features. If we want to obtain a higher prediction performance, our proposed novel metric DISC may be used, the results are obviously improved by using Naïve Bayes and kNN, and even slightly improved by using SVM.

6 Conclusions

Redundant gene elimination is an important topic in the field of bioinformatics. However, the measurement of feature redundancy is still an open problem. Existing metrics including linear and non-linear metrics calculate the redundancy only by the feature's numerical values not considering the label information. Although label information is vital to supervised learning techniques, i.e. classification, the previous works did not use label information to estimate the discriminative contribution of features. Here we propose a novel metric of redundancy based on Discriminative Contribution (DISC) to perform redundant feature elimination. By the new metric, complementary features are likely to be reserved, which is beneficial for the final classification. Experimental results on several microarray data sets show the DISC metric performs better than the commonly used metrics. Redundancy metrics considering the label information are more powerful than those not considering the label information.

This paper only concerns of comparing DISC with the relative redundancy metrics like linear correlation and symmetrical uncertainty in the framework proposed by Yu and Liu [12]. Due to the layout, detailed results of novel algorithms based on the novel metric and their comparison with general feature selection algorithms like Relief and SVM-RFE will be reported in the expanded version.

Acknowledgment

This work was supported in part by the Natural Science Foundation of China under grant no. 20503015, the STCSM "Innovation Action Plan" Project of China under grant no. 07DZ19726, Shanghai Leading Academic Discipline Project under no. J50103, Systems Biology Research Foundation of Shanghai University and Scientific Research Fund of Jiangxi Provincial Education Departments under grant no. 2007-57.

References

1. Schena, M., Shalon, D., Davis, R.W., Brown, P.O.: Quantitative monitoring of gene expression patterns with a complementary dna microarray. Science 270, 467–470 (1995)
2. Golub, T.R., Slonim, D.K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J.P., Coller, H., Loh, M.L., Downing, J.R., Caligiuri, M.A., Bloomfield, C.D., Lander, E.S.: Molecular classification of cancer: Class discovery and class prediction by gene expression. Bioinformatics & Computational Biology 286(5439), 531–537 (1999)
3. Alon, U., Barkai, N., Notterman, D.A., Gish, K., Ybarra, S., Mack, D., Levine, A.J.: Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. Proceedings of the National Academy of Sciences of the United States of America, 6745–6750 (1999)
4. Dudoit, S., Fridlyand, J., Speed, T.P.: Comparison of discrimination methods for the classification of tumors using gene expression data. Journal of the American Statistical Association 97(457), 77–87 (2002)
5. Dougherty, E.R.: Small sample issue for microarray-based classification. Comparative and Functional Genomics 2, 28–34 (2001)
6. Blum, A.L., Langley, P.: Selection of relevant features and examples in machine learning. Artificial Intelligence 97(1-2), 245–271 (1997)
7. Zhou, X., Tuck, D.P.: MSVM-RFE: Extensions of SVM-RFE for multiclass gene selection on DNA microarray data. Bioinformatics 23, 1106–1114 (2006)
8. Ding, C., Peng, H.: Minimum redundancy feature selection from microarray gene expression data. In: Proceedings of the Computational Systems Bioinformatics Conference, pp. 523–529 (2003)
9. Liu, H., Dougherty, E.R., Dy, J.G., Torkkola, K., Tuv, E., Peng, H., Ding, C., Long, F., Berens, M., Parsons, L., Yu, L., Zhao, Z., Forman, G.: Evolving feature selection. IEEE Transaction on Intelligent Systems 20(6), 64–76 (2005)
10. Peng, H., Long, F., Ding, C.: Feature selection based on mutual information: Criteria of max-dependency, max-relevance, and min-redundancy. IEEE Transactions on Pattern Analysis and Machine Intelligence 27(8), 1226–1238 (2005)
11. Yu, L., Liu, H.: Redundancy based feature selection for microarray data. In: Proc. 10th ACM SIGKDD Conf. Knowledge Discovery and Data Mining, pp. 22–25 (2004)
12. Yu, L., Liu, H.: Efficient feature selection via analysis of relevance and redundancy. Journal of Machine Learning Research 5, 1205–1224 (2004)
13. Hall, M.A., Holmes, G.: Benchmarking attribute selection techniques for discrete class data mining. IEEE Transactions on Knowledge and Data Engineering 15(6), 1437–1447 (2003)

14. Guyon, I., Elisseefi, A.: An introduction to variable and feature selection. Journal of Machine Learning Research 3(7-8), 1157–1182 (2003)
15. Forman, G.: An extensive empirical study of feature selection metrics for text classification. Journal of Machine Learning Research 3, 1289–1305 (2003)
16. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers Inc, San Francisco (1993)
17. Press, W.H., Teukolsky, S.A., Vetterling, W.T., Flannery, B.P.: Numerical Recipes in C. Cambridge University Press, Cambridge (1988)
18. Li, J., Liu, H.: Kent ridge bio-medical data set repository (2002), `http://sdmc.lit.org.sg/GEDatasets/Datasets.html`
19. Van't Veer, L.V., Dai, H., Vijver, M.V., He, Y., Hart, A., Mao, M., Peterse, H., Kooy, K., Marton, M., Witteveen, A., Schreiber, G., Kerkhoven, R., Roberts, C., Linsley, P., Bernards, R., Friend, S.: Gene expression profiling predicts clinical outcome of breast cancer. Nature 415(6871), 530–536 (2002)
20. Alizadeh, A.A., Eisen, M.B., Davis, R.E., Ma, C., Lossos, I.S., Rosenwald, A., Boldrick, J.C., Sabet, H., Tran, T., Yu, X., Powell, J.I., Yang, L., Marti, G.E., Moore, T., Jr, J.H., Lu, L., Lewis, D.B., Tibshirani, R., Sherlock, G., Chan, W.C., Greiner, T.C., Weisenburger, D.D., Armitage, J.O., Warnke, R., Levy, R., Wilson, W., Grever, M.R., Byrd, J.C., Botstein, D., Brown, P.O., Staudt, L.M.: Distinct types of diffuse large b-cell lymphoma identified by gene expression profiling. Nature 403, 503–511 (2000)
21. Dietterich, T.G.: Approximate statistical tests for comparing supervised classification learning algorithms. Neural Computation 10, 1895–1923 (1998)

Network-Based Inference of Cancer Progression from Microarray Data

Yongjin Park[1], Stanley Shackney[2], and Russell Schwartz[1]

[1] Department of Biological Sciences, Carnegie Mellon University, 5000 Fifth Avenue, Pittsburgh, PA 15213

[2] Departments of Human Oncology and Human Genetics, Drexel University School of Medicine, West Penn Campus, 320 East North Avenue, Pittsburgh, PA 15212

Abstract. Cancer cells exhibit a common phenotype of uncontrolled cell growth, but this phenotype may arise from many different combinations of mutations. By inferring how cells evolve in individual tumors, a process called cancer progression, we may be able to identify important mutational events for different tumor types, potentially leading to new therapeutics and diagnostics. Prior work has shown that it is possible to infer frequent progression pathways by using gene expression profiles to estimate "distances" between tumors. Individual mutations can, however, result in large shifts in expression levels, making it difficult to accurately identify evolutionary distance from differences in expression. Here, we apply gene network models in order to improve our ability to estimate evolutionary distances from expression data by controlling for correlations among co-regulated genes. We test two variants of this approach, one using full regulatory networks inferred from a candidate gene set and the other using simplified modular networks inferred from clusters of similarly expressed genes. Application to a set of E2F-responsive genes from a lung cancer microarray data set shows a small improvement in phylogenies when correcting from the full network but a more substantial improvement when correcting from the modular network. These results suggest that a network correction approach can lead to better identification of tumor similarity, but that sophisticated network models are needed to control for the large hypothesis space and sparse data currently available.

1 Introduction

One of the most significant insights into cancer biology in recent years has been the discovery that cancers that appear indistinguishable in the clinic may in fact be very different entities at the molecular level [6] with consequently different prognoses and responses to treatment [19,23]. While many possible combinations of genetic abnormalities could theoretically cause the common phenotype of tumor growth, in reality most tumors seem to fall into a limited number of "sub-types" sharing common genetic profiles [13]. Ideally, we would like to know the sequence of mutations responsible for the cancer developing and acquiring increasing aggressiveness in each patient [12,18]. The combination of all such

I. Măndoiu, R. Sunderraman, and A. Zelikovsky (Eds.): ISBRA 2008, LNBI 4983, pp. 268–279, 2008.

sequences of mutations, or "progression pathways," across a population can be summarized in a phylogeny describing the different evolutionary pathways a tumor cell might take in that population. By learning this cancer phylogeny, we hope to better identify common mutational events in tumor formation that can help us develop new diagnostics or therapeutics targeted to specific sub-types.

Desper et al. [3] showed that gene expression measurements can be used to identify progression pathways. The Desper approach assumes that each tumor sample in a population represents one possible progression state, with some tumors likely representing ancestral states of others in the population. By using distances between expression vectors as estimates of evolutionary distance, we can apply standard phylogeny inference algorithms to infer how different tumor states may have evolved in the population. While this phylogenetic approach showed great promise in establishing progression pathways, we would expect it to be partially confounded by the fact that large differences in gene expression profiles need not closely correspond to a large distance in mutational events. Because genes are generally coupled with one another by complicated regulatory networks, a mutation directly altering expression of a gene might affect just that one gene or might indirectly affect a large fraction of expressed genes.

In this paper, we seek to improve our ability to infer progression pathways from microarray data by interpreting expression data in the context of genetic regulatory networks. Our intention is to "correct" for the effects of common regulation when computing expression distances between samples in order to more accurately estimate evolutionary distances between tumors. We accomplish this by inferring Bayesian networks (BNs) describing possible regulatory relationships among genes [5] and using regression models to separate expression changes corresponding to the direct influence of accumulated mutations from indirect expression changes due to regulation by upstream genes. We test two variants of this method, one inferring a full regulatory network from a candidate gene set and the other inferring a simplified modular network structure by first clustering genes with similar expression profiles. We applied both methods to phylogeny inference using a set of E2F-responsive genes extracted from a lung cancer data set. The full network correction generally led to a relatively small improvement in grouping of known clinical sub-types in phylogenies, while the modular network led to a substantially greater improvement. The work establishes that a network correction approach can improve tumor phylogeny inferences, but suggests that care must be taken to deal with sparse and noisy microarray data for these corrections to be reliable.

2 Methods

The input to all of our methods is a microarray data set containing expression levels of a set of genes in a set of samples of tumor cells or healthy tissues. We examine three methods for this inference: a *full network method*, which fits a complete model of a regulatory network to a gene set; a *modular network method*, which collapses genes into a few modules of approximately co-expressed genes prior to network correction; and an *uncorrected method*, which makes inferences

from raw phylogeny data as in Desper et al. [3]. In the uncorrected method, we use Euclidean distances between columns (tumor samples) on the microarray as a distance measure and treat phylogeny inference as a minimum spanning tree (MST) problem. The two correction methods can be regarded as preprocessing filters applied to the raw microarray data in the hope of improving the phylogeny inference.

2.1 Full Network Correction

For the full network method, we begin by inferring a regulatory network among the genes without any prior assumptions about network topology. We perform network inference with a heuristic global optimizer combining Order search ([21]) with conventional local greedy hill climbing. We perform local optimization using the L1-regularized Markov Blanket ([16]) method (L1MB), a technique for pruning uninformative edges from an initial candidate edge set. We assume we are given a set of nodes V and a set of expression measurements X, where for any $v_i \in V$, X_{v_i} is the vector of expression values of node v_i across genes or modules. The method learns a BN $\mathcal{G} = (V, E)$ with edges $\forall i, (v_{pa(i)}, v_i) \in E$ by fitting a set of regression equations of the form

$$X_{v_i} = w_{i,pa(i)}^T X_{v_{pa(i)}} + \epsilon$$

with $\epsilon \sim \mathcal{N}(0, \Sigma)$. Fitting the model involves learning a set of weight values of the form $w_{i,pa(i)}$, each corresponding to the strength of the inferred regulatory influence of some parent node $pa(i)$ on a child node v_i by the linear Gaussian model. These weights, and thus the connectivity of the network, can be found by solving the following quadratic programming problem:

$$\underset{w}{\text{argmin}} \quad \sum_x (X_{v_i} - w_{i,pa(i)}^T X_{v_{pa(i)}})^2 \quad \text{subject to} \quad ||w||_1 < \mathbf{t} \tag{1}$$

The full global optimizer was run for 50 random restarts for each data set. For each regularization step, we computed the Bayesian Information Criterion (BIC) score and selected the one best model at the conclusion of the heuristic global optimization. The result of this stage is a network defining parent-child regulatory relationships among genes and a set of regression coefficients estimating the strength of each such relationship.

Once we have learned a network, we can use that network to produce a corrected set of expression data. For each row of our input matrix X_{v_i}, corresponding to uncorrected expression of gene i across samples, we produce a corrected row Y_{v_i} as follows:

$$Y_{v_i} = \sum_{pa(i) \in V} X_{v_i} - w_{i,pa(i)} X_{v_{pa(i)}}$$

That is, we subtract from each expression value of v_i the portion of that value attributed to each parent $pa(i)$ by the regression model learned above. The corrected matrix Y then becomes the input to the phylogeny inference step.

2.2 Modular Network Correction

With the modular network method, our first step is identifying modules (clusters) of approximately co-expressed genes within the data. We use a Dirichlet process mixture (DPM) [1,11,15], a non-parametric mixture model chosen because it does not require prior knowledge of the number of modules to be discovered. We can understand how the model works by considering how a single gene might be assigned to a module. Suppose we have so far assigned $M^* - 1$ genes to K^* modules. The probability of M^*-th gene belonging to one of the K^* currently known modules is given by

$$p(c^k_{M^*} = 1 | c_{1:M^*-1}) = \frac{\sum_m^{M^*-1} c^k_m}{M^* - 1 + \alpha}, \tag{2}$$

where $c_m \sim DP(\alpha)$. Alternatively, the M^*-th gene could be the first member of a newly generated $K^* + 1$-th module. The probability of this event is given by

$$p(\forall m \in [1..M^* - 1], c^k_{M^*} \neq c^k_m | c_{1:M^*-1}) = \frac{\alpha}{m^* - 1 + \alpha}. \tag{3}$$

This module assignment prior probability (Eq. 2, 3) can be weighted by each component's marginal data likelihood. These probability formulas multiplied over all gene assignments define the probability of any possible assignment of genes to modules. Using the collapsed Gibbs sampling scheme, we can exactly sample all the module assignments (reviewed in [11]). For this work, we utilized an implementation of the Chinese Restaurant process [14], one possible realization of the Dirichlet process model, which considers each microarray experiment as one Gaussian distribution.

Once we learn gene modules, we infer a Bayesian network across the modules using a regression method similar to that used for the full network correction model. We begin, however, with a restricted network topology defined by the module structure. We tested several possible methods for restricting network topologies, developing reduced representations of network expression, and learning regulation between modules, with the results reported here reflecting what we judged to be the best balance of accuracy and robustness observed. Our chosen method assumes an initial candidate edge set that ignores regulation within co-expression modules but allows full connections between modules. Let $x^{(n)}_{i,m}$ denote n^{th} observation of i^{th} gene in m^{th} module. Then we can learn relationships from all the genes in m'^{th} module toward this gene by posing a regression model similar to that of the full network method:

$$X_{v_i \in m} = \sum_{j \in m'} w_j X_{v_j} + \epsilon \tag{4}$$

where w_j is a weight coefficient and ϵ is an additive error term. To optimize edge coefficients, we perform a lasso regression where we optimize weight vectors by the following formula:

$$\mathbf{w} \leftarrow \underset{w}{argmin} \sum_n^N \left(X^{(n)}_{v_i \in m} - \sum_j w_j x^{(n)}_{v_j \in m'} \right)^2 \quad \text{subject to} \quad \sum_j |w_j| < \lambda \tag{5}$$

using the BIC metric to select the best λ. This lasso regression approach provides a way of enforcing sparsity of the final model.

Once the network model is inferred, the modular network method corrects expression values exactly as does the full network model, converting the raw expression matrix X into a corrected matrix Y using the previously inferred weight coefficients for the correction.

2.3 Phylogeny Inference

All three methods end with a common phylogeny inference step, with this step constituting the entirety of the uncorrected method. We construct tumor phylogenies using a simple variant of the method of Desper et al. [3]. The input to this stage is a matrix of expression values in which columns correspond to tumor samples and rows correspond to genes. We first compute pairwise distances between samples using Euclidean distances between the expression vectors of the two samples. The set of pairwise distances establishes a complete, weighted graph whose nodes are the samples. We define the most plausible phylogeny on the tumors to be the minimum spanning tree (MST) on the graph. We find the MST using a subroutine in the Bayes Net Toolbox for Matlab [10]. Note that we do not infer Steiner nodes (unobserved nodes that allow for reduced phylogeny cost relative to the MST). Optimally solving the Steiner node inference problem is computationally intractable, and experiments with heuristic Steiner node inferences, omitted here due to space limitations, produced only marginal improvements in quality over the MST solutions.

2.4 Validation

We compared the three methods using a set of microarray expression data from 72 lung cancer samples and 19 healthy controls [7]. We retrieved the data from the Entrez Gene Expression Omnibus database (`http://www.ncbi.nlm.nih.gov/projects/geo/`) entry GSE1037. Each sample was labeled by the submitters with one of seven clinical sub-types: normal cell, adenocarcinoma, primary typical carcinoid, large cell neuroendocrine carcinoma, primary large cell lung carcinoma, primary small cell lung carcinoma, and primary combined small cell lung carcinoma-adenocarcinoma. We grouped combined small cell lung carcinoma-adenocarcima with primary small cell carcinomas but otherwise treated these submitter labels as the true class labels in our subsequent validation. We restricted our analysis to a set of candidates genes likely to be significant to multiple tumor cell types. For this purpose, we chose a set of genes believed to be downstream of E2F-family transcription factors, because of the importance of the E2F regulatory network to a broad set of cancers [22,4]. We manually selected a subset of 325 genes reported in the literature to be E2F-responsive, excluding those reports based only on a single microarray result. The resulting set consisted of genes identified by PCR, Northern blot analysis, or at least two independent microarray studies. Of these 325 candidate genes, 278 were present on the Jones et al. microarray and were used in this study.

Each of the methods described above — uncorrected, full network, and modular network — was applied to the resulting set of 278 genes in 91 samples.

We measured overall phylogeny quality by assuming that the clinical subtype labels assigned by Jones et al. correspond to true evolutionary classes of tumors and that tumor samples with the same labels should therefore be near one another in the phylogeny. We quantified the ability of the phylogeny to group samples with a given label by the mean square number of edges separating pairs of samples with that label in the phylogeny. To better quantify differences between the trees, we further analyzed the network topologies by using the ratio of that value to the mean square number of edges separating pairs of samples with distinct labels in the phylogeny. These values were computed in Matlab by the Floyd-Warshall algorithm [2].

In addition to testing accuracy in grouping class assignments in the phylogenies, we were interested in robustness of phylogeny inferences to missing data. Robustness gives us a measure of the reliability of specific edges in the phylogenies. We assessed robustness by the presence or absence of edges between specific pairs of samples for inferences from the full data set versus inferences based on a randomly selected subset of the genes. We use these edge comparisons to identify false positive and false negative error rates for each method as follows:

$$err_{FP} = \frac{\sum_{e\in(i,j)}(1-\delta(e))\hat{\delta}(e)}{\sum_{e\in(i,j)}\hat{\delta}(e)} \quad \text{and} \quad err_{FN} = \frac{\sum_{e\in(i,j)}\delta(e)(1-\hat{\delta}(e))}{\sum_{e\in(i,j)}(1-\hat{\delta}(e))}, \quad (6)$$

where $\delta \in \{0,1\}$ denotes true edge indicator and $\hat{\delta}$ denotes inferred edge indicator. This test was repeated for each method for inferences using 10%-90% of the full gene set, in increments of 10%, with 100 random replicates for each fraction.

3 Results

We ran each of the three methods on our full data set, producing one phylogeny per method. Fig. 1(a) shows the inferred phylogeny for the uncorrected method, Fig. 1(b) the phylogeny produced by the full network method, and Fig. 1(c) the phylogeny produced by the modular network method. All three share some common features, although there are also significant rearrangements among them. All three trees predominantly place the adenocarcinomas in a cluster adjacent the normal cells, suggesting that the adenocarcinomas may be similar to an early progression state. The three trees differ on the children of the adenocarcinomas, though. The uncorrected tree links neuroendocrine tumors below adenocarcinomas. The fully corrected tree, however, places the carcinoid tumors below the adenocarcinomas. In the modular tree, the adenocarcinomas form a leaf sub-tree, not leading to any other large group. All three trees have a subtree containing neuroendocrine tumors above small-cell tumors. In the uncorrected and full network trees, large cell tumors also for the most part branch off of the neuroendocrines in this sub-tree while in the modular tree, the large cell tumors are not well grouped into a single sub-tree. The carcinoid tumors largely form

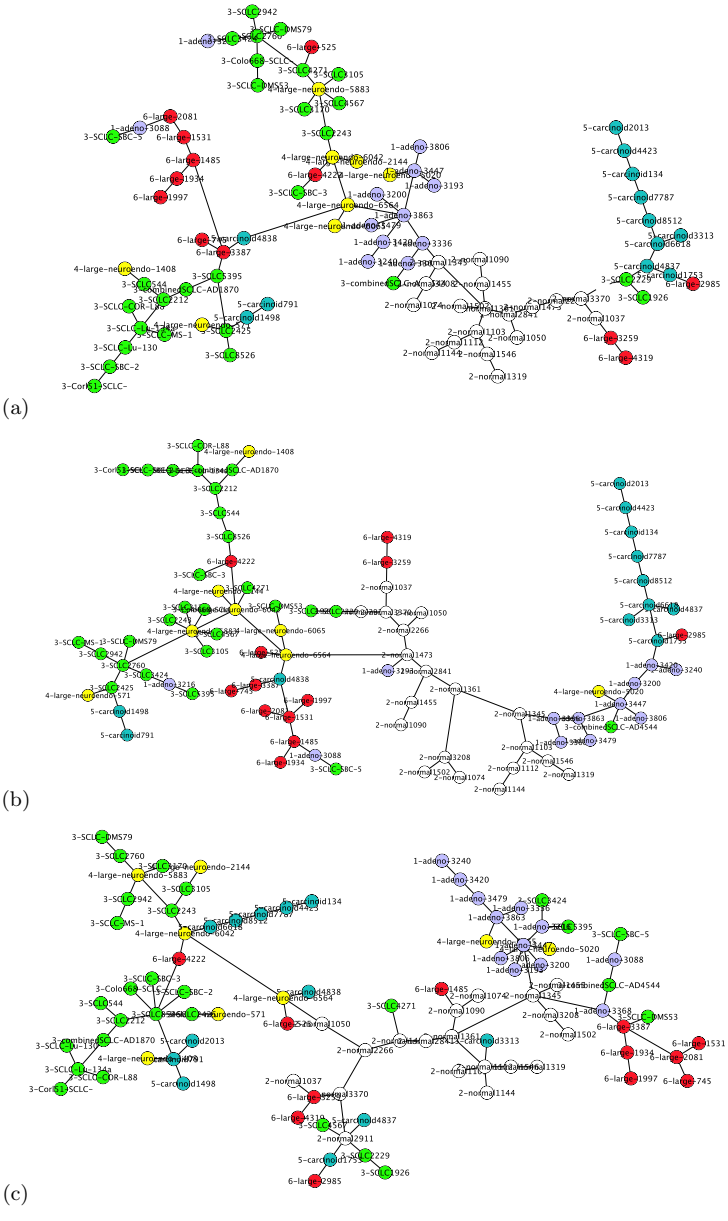


Fig. 1. Cancer phylogenies inferred from E2F-responsive genes in lung cancer. Each node in the graph corresponds to a single tumor sample and each edge to an inferred phylogenetic relationship between two tumors. Nodes are labeled with IDs from the Entrez GEO database and colored to identify different tumor sub-types: normal cells (white), large cell (red), large cell neuroendocrine (yellow), small cell (green), typical carcinoid (blue), and adenocarcinoma (purple). (a) Uncorrected method; (b) Full network method; (c) Modular network method.

a tightly packed, nearly linear chain, yet the placement of this chain is different for each method. The uncorrected tree places the carcinoids as a separate progression pathway off of normal cells, the full network tree places them as a child sub-tree of adenocarcinomas, and the modular network tree places them as a child sub-tree of neuroendocrine tumors.

We assessed the quality of the trees by their ability to closely group samples with common labels. On a visual examination, none of the trees is obviously superior to the others by this criterion. We therefore performed a quantitative assessment using mean square distances between samples with common class assignments. Fig. 2(a) shows the results for each of the labels. The full network method shows generally somewhat improved clustering relative to the uncorrected data, with better clustering of some labels at the expense of others. The modular network method overall shows approximately twice as much improvement as the full network method relative to the uncorrected method, although it too exhibits poorer performance on some classes than the uncorrected method. Relative to the uncorrected method, the full network method performs most poorly on classes 1 (adenocarcinoma) and 4 (neuroendocrine), with slightly worse performance on class 2 (normal cells). It achieves overall better performance predominantly by large improvements on classes 3 (small cell) and 6 (large cell). The modular tree also has difficulty with class 4 (neuroendocrine), but does similarly to the uncorrected method on 2 (normal cells) and 3 (small cell) and much better on 1 (adenocarcinoma), 4 (neuroendocrine), and 6 (large cell).

A visual examination of the phylogenies suggested that the modular method may do better by this metric in part because it tends to produce a more compact graph, with a larger number of high-degree hubs predicted to be common ancestors of many independent branches. To test that intuition, we also examined the ratio of mean square distance within each tumor class to mean square distance between classes. Fig. 2(b) confirms that when normalized for mean distance between classes, all three measures show similar results. There are very slight differences between the trees when assessed with this normalization, with the uncorrected tree marginally superior to the full network tree, and the full network tree marginally superior to the modular network tree. The relative performances of the different methods on the individual tumor classes are largely unchanged relative to the unnormalized data in Fig. 2(a), but the advantages of both correction methods on some sub-classes are significantly reduced. This result confirms the hypothesis that the modular method, and to a lesser degree the full correction method, produce good clustering of clinical sub-types in part by biasing the overall graph more towards a compact "hub-and-spoke" topology.

We then further examined robustness of the three methods in order to assess how much each method suffers from the limited data available. Fig. 3 shows false positive and false negative rates for specific inferred edges in the phylogenies as we reduce the number of genes available to the inferences. The plot shows that the robustness of the methods, as measured by their ability to reliably reproduce the same tree in the presence of reduced data sets, decreases with the correction methods despite the increase in tree quality with the correction methods. The

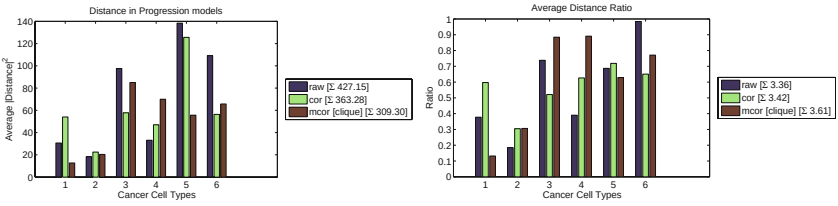


Fig. 2. Assessment of phylogeny quality based on clustering of common labels. (a) Mean square distance on the phylogeny for tumors sharing each label for each of the three methods. (b) Ratio of mean square distance for tumors sharing each label to mean square distance for tumors with distinct labels for each of the three methods. The legends provide accumulated scores across all labels for each method. The labels are (1) adenocarcinoma, (2) normal, (3) small cell, (4) large cell neuroendocrine, (5) carcinoid, and (6) large cell.

modular method shows the least robustness of the three, with the full network model slightly better, and the uncorrected the best of all. Absolute robustness is similar for all three methods, though, suggesting that the corrections have only a modest negative impact on sensitivity to sparse data.

We can gain some understanding of where the methods might be improved by examining the results of the network inferences themselves. Fig. 4 shows the inferred network topology for the full network model. The obvious question we wish to ask about such a network is whether it actually corresponds to regulatory relationships among the E2F-responsive genes. To answer this question, we used BiNGO [9], an open-source Java tool for evaluating the significance of gene clusters based on common Gene Ontology categories. BiNGO identified a meaningful relationship for only one component of Fig. 4, highlighted in red and yellow. This component is overrepresented for several GO categories related to the cell cycle, most significantly the M phase (GO:0000279) with p-value 0.00025. We can conclude that the while the network is not random, it does not

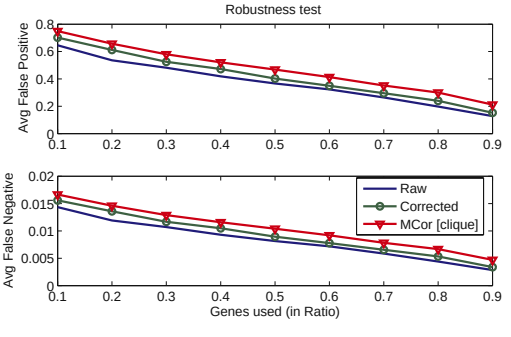


Fig. 3. Robustness of phylogeny inferences. The plots show false positive and false negative rates for edge assignments relative to the full-data inferences as a function of fraction of genes examined.

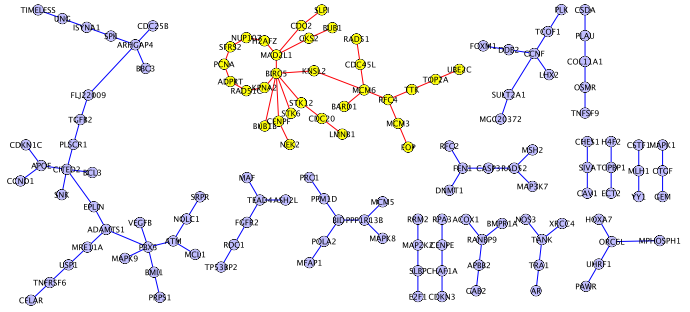


Fig. 4. Inferred E2F-responsive network topology from the full-network model. Genes that were inferred to have no neighbors in the model were omitted from the diagram. A sub-network enriched for cell cycle-related genes is highlighted in yellow and red.

for the most part correspond well with known regulatory relationships among these genes. This result suggests that a major reason the full network method does not perform better is that the network inference step is unable to learn the true network reliably, an issue we consider further in the Discussion.

4 Discussion

We have examined the utility of network-based correction methods for improving our ability to infer evolutionary relationships between tumor types based on expression data. A simple correction based on a full inferred gene network appears to provide a small improvement in phylogeny inference relative to the uncorrected approach. A modular network approach that simplifies the network inference problem by clustering genes prior to network inference leads to a substantially greater improvement in phylogeny accuracy. It therefore does appear that a network-based correction method can improve our ability to infer evolutionary relationships between tumor types, but some sophistication is needed in the method to ensure that the network can be learned from the available data. These improvements in quality paradoxically come at the cost of some loss in robustness, suggesting that further care may be needed to avoid increasing noise sensitivity when reducing the confounding effects of gene regulation.

We believe that the full network method performs comparatively poorly primarily because its network inferences are insufficiently accurate. Network inference from raw expression data is well known to be a difficult problem. Exploiting the modular structure of expression networks may reduce the solution space leading to more accurate inferences from limited data, as suggested by the results of Segal et al. [17]. A common approach for dealing with sparse expression data in practice is to use many heterogeneous data sources — such as predicted transcription factor binding motifs or direct measurements of transcription factor binding — in order to improve accuracy (see, for example, Tavazoie et al. [20]). Our full network approach might similarly benefit from additional data sources.

Alternatively, we might use literature-derived networks in place of automated network inferences. It is also possible that the linear regression method used for correction within the method is inadequate for describing non-linear relationships between genes and that a more versatile non-linear regression model, such as that of Kim et al. [8], might alleviate this problem. Our conclusions about network correction may also be overly pessimistic because of our decision to use a small, curated gene set rather than uncurated whole-genome expression data that would likely benefit more from correction.

While the goal of this study was to learn about progression among cancers, it is difficult to identify specific features of the inferred tree topologies in which we can have high confidence. Some features are highly robust to the method chosen and may therefore be more reliable, such as the inferences that the adenocarcinomas resemble an early progression stage off of normal cells and that the neuroendocrine tumors resemble an ancestral state of small cell tumors. There remains at present no independent method to verify such predictions, though, and these conclusions are for the moment merely speculative. An important direction in future work will be determining whether inferred tree topologies are robust to different data sets and whether a more detailed study of the molecular profiles of these tumor types supports our predicted tumor progression pathways.

Acknowledgments

This work was supported in part by the Eberly Family and U.S. National Science Foundation award #0612099.

References

1. Antoniak, J.R.: Mixtures of Dirichlet processes with applications to Bayesian nonparametric problems. Annals Stat. 2, 1152–1174 (1974)
2. Cormen, T.H., Leiserson, C.A., Rivest, R.L., Stein, C.: Introduction to Algorithms. MIT Press, Cambridge (2001)
3. Desper, R., Khan, J., Schaffer, A.A.: Tumor classification using phylogenetic methods on expression data. J. Theor. Biol. 228, 477–496 (2004)
4. Fang, Z.H., Han, Z.C.: The transcription factor E2F: a crucial switch in the control of homeostasis and tumorigenesis. Histol. Histopathol. 21, 403–413 (2006)
5. Friedman, N., Linial, M., Nachman, I., Pe'er, D.: Using Bayesian networks to analyze expression data. J. Comput. Biol. 7, 601–620 (2000)
6. Golub, T.R., Slonim, D.K., Tamayo, P., Huard, C., Gaasenbeej, M., Mesirov, J.P., Coller, H., Loh, M.L., Downing, J.R., Cligiuri, M.A., Bloomfield, C.D., Lander, E.S.: Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. Science 286, 531–537 (1999)
7. Jones, M.H., Virtanen, C., Honjoh, D., Miyoshi, T., Satoh, Y., Okumura, S., Nakagawa, K., Nomura, H., Ishikawa, Y.: Two prognostically significant subtypes of high-grade lung neuroenedocrine tumours independent of small-cell and large-cell neuroendocrine carcinomas identified by gene expression profiles. Lancet 363, 775–781 (2004)

8. Kim, S., Imoto, S., Miyano, S.: Dynamic Bayesian network and nonparametric regression for nonlinear modeling of gene networks from time series gene expression data. Biosystems 75, 57–65 (2004)
9. Maere, S., Heymans, K., Kuiper, M.: BiNGO: A Cytoscape plugin to assess overrepresentation of Gene Ontology categories in biological networks. Bioinformatics 21, 3448–3449 (2005)
10. Murphy, K.: Bayes net toolbox for Matlab (2007), `http://www.cs.ubc.ca/~murphyk/Software/BNT/bnt.html`
11. Neal, R.M.: Markov chain sampling methods for Dirichlet process mixture models. J. Comput. Graph. Stat. 9(2), 249–265 (2000)
12. Nowell, P.C.: The clonal evolution of tumor cell populations. Science 194, 23–28 (1976)
13. Perou, C.M., Sorlie, T., Eisen, M.B., van de Rijn, M.M., Jeffrey, S.S., Rees, C.A., Pollack, J.R., Ross, D.T., Johnsen, H., Akslen, L.A., Fluge, O., Pergamenschikov, A., WIlliams, C., Zhu, S.X., Lonning, P.E., Borresen-Dale, A.-L., Brown, P.O., Botstein, D.: Molecular portraits of human breast tumors. Nature 406, 747–752 (2000)
14. Qin, Z.S.: Clustering microarray gene expression data using weighted Chinese restaurant process. Bioinformatics 22(16), 1988–1997 (2006)
15. Rasmussen, C.E.: The infinite Gaussian mixture model. In: Solla, S.A., Lean, T.K., Muller, K.-R. (eds.) Advances in Neural Information Processing Systems, vol. 12, pp. 554–560. MIT Press, Cambridge (2000)
16. Schmidt, M., Niculescu-Mizil, A., Murphy, K.: Learning graphical model structure using L1-regularization paths. In: Proceedings of the 22nd Conference on Artificial Intelligence (AAAI 2007) (2007)
17. Segal, E., Shapira, M., Regev, A., Pe'er, D., Botstein, D., Koller, D., Friedman, N.: Module networks: identifying regulatory modules and their condition-specific regulators from gene expression data. Nat. Genet. 34(2), 166–176 (2003)
18. Shackney, S.E., Silverman, J.F.: Molecular evolutionary patterns in breast cancer. Anat. Pathology 10, 278–290 (2003)
19. Sorlie, T., Perou, C.M., Tibshirani, R., Aas, T., Geisler, S., Johnsen, H., Hastie, T., Eisen, M.B., van de Rijn, M., Jeffrey, S.S., Thorsen, T., Quist, H., Matese, J.C., Brown, P.O., Botstein, D., Lonning, P.E., Borresen-Dale, A.-L.: Gene expression patterns of breast carcinomas distinguish tumor subclasses with clinical implications. Proc. Natl. Acad. Sci. USA 98, 10869–10874 (2001)
20. Tavazoie, S., Hughes, J.D., Campbell, M.J., Cho, R.J., Church, G.M.: Systematic determination of genetic network architecture. Nature Genet 22, 281–285 (1999)
21. Teyssier, M., Koller, D.: Ordering-based search: A simple and effective algorithm for learning Bayesian networks. In: Proceedings of the 21th Annual Conference on Uncertainty in Artificial Intelligence (UAI-2005), pp. 584–559 (2005)
22. Tsantoulis, P.K., Gorgoulis, V.G.: Involvement of E2F transcription factor family in cancer. Eur. J. Cancer 41, 2403–2413 (2005)
23. van 't Veer, L., Dai, H., van de Vijver, M., He, Y., Hart, A., Mao, M., Peterse, H., van der Kooy, K., Marton, M., Witteveen, A., Schreiber, G., Kerkhoven, R., Roberts, C., Linsley, P., Bernards, R., Friend, S.: Gene expression profiling predicts clinical outcome of breast cancer. Nature 415(6871), 530–536 (2002)

Invited Keynote Talk: Quiet Revolution: Connectivity in the Cancer Research Community

Kenneth Buetow

National Cancer Institute Center for Bioinformatics
Rockville, Maryland
holderma@mail.nih.gov

Biomedicine in general - and cancer research especially - are moving toward an enhanced understanding of the molecular basis of disease. To realize the benefits of this revolution - earlier detection, more productive drug discovery and development, more individualized patient care - biomedical researchers need to utilize all the new technologies available in basic and clinical research. The National Cancer Institute has developed the caBIG™ initiative (cancer Biomedical Informatics Grid™) to overcome the "silos" and data disconnects that slow cancer research. This voluntary, open-source network is unique in the history of biomedical research. caBIG™ connects scientists and practitioners through a shareable, interoperable infrastructure; provides standard rules, unified architecture, and a common language to share information; and offers free, open-source software tools for collecting, analyzing, integrating, and disseminating information. With that foundation, NCI is fulfilling the vision of a truly connected cancer enterprise across the nation that achieves optimum research productivity for improved clinical outcomes.

I. Măndoiu, R. Sunderraman, and A. Zelikovsky (Eds.): ISBRA 2008, LNBI 4983, p. 280, 2008.

Wavelet-Based 3-D Multifractal Spectrum with Applications in Breast MRI Images

Gordana Derado[1], Kichun Lee[3], Orietta Nicolis[4], F. DuBois Bowman[1], Mary Newell[2], Fabrizio F. Rugger[5], and Brani Vidakovic[3]

[1] Emory University, Atlanta, GA
[2] Winship Cancer Institute, Atlanta, GA
[3] Georgia Institute of Technology and Emory University, Atlanta, GA
[4] University of Bergamo, Italy
[5] CNR Milano, Italy

Abstract. Breast cancer is the second leading cause of death in women in the United States. Breast Magnetic Resonance Imaging (BMRI) is an emerging tool in breast cancer diagnostics and research, and it is becoming routine in clinical practice. Recently, the American Cancer Society (ACS) recommended that women at very high risk of developing breast cancer have annual BMRI exams, in addition to annual mammograms, to increase the likelihood of early detection. (Saslow *et al.* [20]). Many medical images demonstrate a certain degree of self-similarity over a range of scales. The multifractal spectrum (MFS) summarizes possibly variable degrees of scaling in one dimensional signals and has been widely used in fractal analysis. In this work, we develop a generalization of MFS to three dimensions and use dynamics of the scaling as discriminatory descriptors for the classification of BMRI images to benign and malignant. Methodology we propose was tested using breast MRI images for four anonymous subjects (two cancer, and two cancer-free cases). The dataset consists of BMRI scans obtained on a 1.5T GE Signa MR (with VIBRANT) scanner at Emory University. We demonstrate that meaningful descriptors show potential for classifying inference.

1 Introduction

In the United States, breast cancer is the second leading cause of death in women (after lung cancer), and is the most common cancer among women. One out of eight women will develop breast cancer in their lifetime. The American Cancer Society (ACS) estimated that about 40,460 women would die from the disease in 2007 (Jemal *et al.* [10]). Studies have indicated that early detection and treatment improve the chances of survival for breast cancer patients (Curpen *et al.* [6], Smart *et al.* [21]). Breast imaging plays a vital role in screening for and diagnosis of breast cancer and in monitoring the impact of treatment. In this study, we target the development of analytical techniques to improve diagnostic capabilities of BMRI.

I. Măndoiu, R. Sunderraman, and A. Zelikovsky (Eds.): ISBRA 2008, LNBI 4983, pp. 281–292, 2008.

While mammography and breast ultrasound are considered "gold standard" for breast cancer screening, an increasing body of research has shown BMRI to be an effective diagnostic and interventional tool. BMRI has been approved by the *U.S. Food and Drug Administration* since 1991 for use as a supplemental tool to mammography for breast cancer diagnostics. It is also useful in breast cancer staging, in treatment and preoperative planning, and for patient follow-up after breast cancer treatment. Since 1999, there has been a 40% per year increase in the number of BMRI examinations in the United States. Recently, ACS recommended that women at very high risk of developing breast cancer have annual BMRI exams, in addition to annual mammograms, to increase the likelihood of early detection (Saslow *et al.* (2007) [20]).

Based on the principles of nuclear magnetic resonance (NMR), a technique that is highly sensitive to physical, chemical and biological characteristics of tissues and fluids, BMRI enables a 3-D examination of breast tissue and provides a noninvasive assessment of the microcirculatory characteristics of tissues, in addition to traditional anatomical information. The 3-D anatomical structure, is insufficient for distinguishing between benign and malignant tissues, and functional imaging is typically incorporated. In this setting, functional imaging utilizes contrast agents for MRI, which enables the visualization of functional changes when serial MRI scans are acquired. The typical contrast agent for BMRI is Gadolinium (GAD) diethylenetriamine penta-acetic acid (DTPA).

Evaluating BMRI accurately and efficiently is essential, but it is very challenging in practice. BMRI produces massive 4-dimensional (three spatial dimensions plus a time dimension) data, posing challenges for analysis and detection. At present, BMRI cannot always distinguish between cancerous and non-cancerous functional dynamics, prompting the investigation into improved methods.

Wavelet techniques have become indispensable for image processing, in particular when dealing with medical images. Mallat's multiresolution analysis (see Vidakovic [22]) decomposes an image into a set of approximation coefficients (low frequency components) and the scale dependent hierarchy of detail coefficients (high frequency components). A standard tensor product orthogonal wavelet transformation of an image results in three sets of generated detail coefficients: diagonal, horizontal and vertical. Numerous references can be found in the literature in which wavelets are applied to mammogram images. For example, in Yoshida *et al.* [23], a wavelet transform was applied to detect clustered microcalcifications. In Zheng *et al.* [24] and Derado *et al.* [5], a wavelet-based image-enhancement method is employed to enhance microcalcification clusters for improved detection. Recently there has been an increase in the use of wavelet-based methodology in the analysis of BMRI data. Alterson and Plewes (2003)[1] used a multiresolution non-orthogonal wavelet representation as a measure of similarity to detect natural biological symmetries in breast MRI scans. Mainardi *et al.* (2007) [13] present a nonrigid registration algorithm of dynamic MR breast images based on a multiresolution motion estimation of the breast using complex discrete wavelet transform. To the best of our knowledge, however, approaches

using scaling methodology in BMRI data can not be found in the published literature.

Fractality is a concept pervasive in medical research. Many medical signals and images demonstrate a certain degree of self-similarity over a range of scales, lending to the development of algorithms based on fractal analysis of those objects (see Chen *et al.* [4] and Kuklinski [12]). For example, fractality was used to detect breast cancer in Priebe *et al.* [16], Kestener *et al.* [11], and Bocchi *et al.* [2]. Chen *et al.* [4] developed a pattern recognition technique based on features derived from the fractal description of mammograms. Kuklinski [12] used a wavelet transform modulus maxima method generalized to the two dimensional case. They combined this approach with a multifractal analysis, enabling the detection of tumors as well as microcalcifications. Kestener *et al.* [11] used long range correlations and wavelet-based multifractality for tissue classification in digitized mammograms to support clinical diagnosis. In Moloney *et al.* [14], the MFS is used to analyze the pupillary behavior of older adults and to discriminate between patients with various ocular acuity.

Processes with fractal characteristics that exhibit rich scaling behavior are often referred to as *multifractals*. The fractional Brownian motion (fBm), a theoretical model for mono-fractality, is a non-stationary process whose sample paths exhibit a homogeneous degree of regularity. For many applications, this homogeneous regularity may be too restrictive. In particular, one may want models that account for differing degrees of regularity. Multifractal analysis is concerned with describing the local singular behavior of functions in a geometrical and statistical fashion. It was first introduced in the context of turbulence and applied in many other contexts such as Diffusion Limited Aggregation (DLA) patterns research, earth quake distribution analysis, signal processing and internet data traffic modeling. For an introduction to multifractals, see Riedi [18]. Multifractal models exhibit patterns of locally varying scaling behavior similar to that encountered in medical and biological data (among others). They usually exhibit a prevalent scaling behavior, but a multitude of other scalings may also be present although occurring much less frequently. Since multifractal models are in general non-stationary, standard tools in time series analysis such as the Fourier transform are not appropriate because the Fourier transform is not localized in time. Evaluating the varying local properties of multifractal processes requires analytical methods that are able to localize information in time and frequency. Given that wavelets are local in both frequency/scale (via dilations) and in time (via translations), the wavelet defined multiscale analysis is convenient in assessing multitude scalings intrinsic for BMRI scans. For a detailed study of multifractals, we refer the reader to Riedi [17] and Morales [15].

The multifractal spectrum (MFS) summarizes possibly variable degrees of scaling in signals. In the case of fractals, scaling refers to the propagation of energy when the signals or images are inspected at various resolutions. The dynamics of the scaling can be used as discriminatory descriptors; thus, multifractality provides an additional window through which to look at the data and renders standard statistical approaches insufficient.

In this work, we generalize the concept of multifractal spectrum as it was defined in Gonçalves *et al.* [9] to the three dimensional case and use some of its low-dimensional descriptors to classify BMRI scans as either benign or malignant. Although the number of subjects analyzed is small (two cases and two controls), our findings are consistent with empirical evidence that healthy responses are characterized by irregularity and that increased regularity may suggest pathologies.

The paper is organized as follows. Section 2 gives a description of the data to which we apply our proposed method. In Section 3, we provide a brief review of the theoretical background of wavelets. In addition, the three dimensional multifractal spectrum is defined and some of its properties are illustrated on the example of 3-D fractional Brownian motion MFS. Section 4 deals with the application of our proposed methodology to cancer detection via the classification of BMRI. In Section 5 we provide conclusions and delineate some possible directions for future research.

2 Description of the Data

The data consist of serial BMRI scans from each of four women: 2 cancer and 2 cancer-free cases. The scan series includes one pre-contrast image and four post-contrast images acquired at 1, 3, 5, and 7 minutes after the contrast is administered. The discriminatory pattern of contrast enhancement, characterized by rapid accumulation in the malignant mass and rapid wash out, occurs in the first few minutes following injection. By 7 minutes or later, the contrast uptake in most breast tissue is enhancing. Each 3-D scan contains 104 sagittal slices comprised of an array of 256×256 pixels and slice thickness of 3mm. The scans were obtained on a 1.5T GE Signa MR (with VIBRANT) scanner at Emory University.

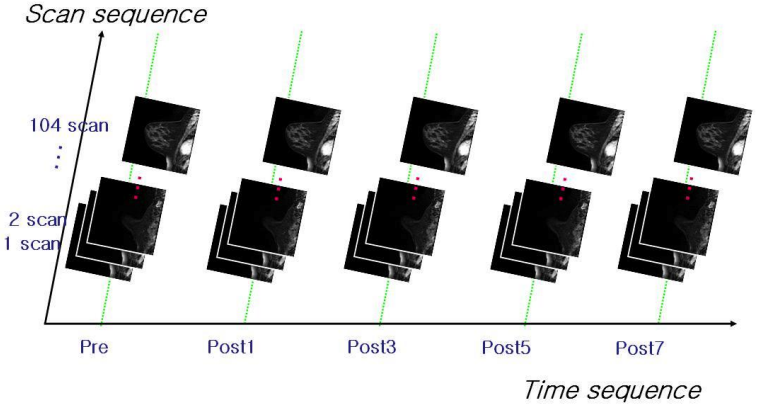


Fig. 1. Illustration of the data structure and acquisition

3 Methodology

In this work we present a conceptual description of MFS in three dimension and demonstrate its utility in the classification of BMRI images. Our approach consists of two main steps. First, we estimate the multifractal spectra and extract a number of low dimensional summaries (such as slopes, tangents, broadness and spectral mode). Then, we use these summaries as discriminatory measures for BRMI images.

3.1 Background on Wavelets

The 3-D wavelet basis functions are constructed via translations and dilations of a tensor product of univariate wavelets and scaling functions. For technical reasons, we consider L^1-normalization of wavelets instead of standard L^2 normalization, of which expression for $\psi_{j,\mathbf{k}}, \phi_{j,\mathbf{k}}$ is,

$$\phi_{j,\mathbf{k}}(\mathbf{x}) = 2^{3j}\phi(2^j x_1 - k_1, 2^j x_2 - k_2, 2^j x_3 - k_3)$$
$$\psi^i_{j,\mathbf{k}}(\mathbf{x}) = 2^{3j}\psi^i(2^j x_1 - k_1, 2^j x_2 - k_2, 2^j x_3 - k_3)$$

where $i = h, l, v, hl, hv, lv, hlv$ denote the different directions on a cube (see Fig. 2, left), $\mathbf{x} = (x_1, x_2, x_3) \in \mathbf{R}^3$, and $\mathbf{k} = (k_1, k_2, k_3) \in \mathbf{Z}^3$. Then, any function $f \in \mathbf{L}_2(\mathbf{R}^3)$ can be represented as

$$f(\mathbf{x}) = \sum_{\mathbf{k}} c_{j_0,\mathbf{k}}\phi_{j_0,\mathbf{k}}(\mathbf{x}) + \sum_{j \geq j_0}\sum_{\mathbf{k}}\sum_{i} d^i_{j,\mathbf{k}}\psi^i_{j,\mathbf{k}}(\mathbf{x}) \tag{1}$$

where the wavelet coefficients are given by

$$d^i_{j,\mathbf{k}} = 2^{3j}\int f(\mathbf{x})\psi^i(2^j\mathbf{x} - \mathbf{k})\mathrm{d}\mathbf{x} \tag{2}$$

The 3-D multifractal wavelet spectra will be defined using the wavelet coefficients $d^i_{j,\mathbf{k}}$, along the scale index j. We assume that the mother wavelet ψ has $\mathcal{R}$ vanishing moments, that is, $\int x^r\psi(x)dx = 0, r = 0, \ldots, \mathcal{R}$, because the decorrelation property of wavelet coefficients depends upon this assumption.

Although the wavelet analysis of n-dimensional structures is conceptually straightforward, it is not routinely implemented in standard wavelet software and for this project we developed and implemented the three dimensional transformation in MATLAB's freely available package Wavelab [3].

3.2 3-D Multifractal Spectrum

In Gonçalves *et al.* [9], it is shown how the oscillatory or scaling behavior of a process carries over into the local scaling properties of its wavelet coefficients $d^i_{j,\mathbf{k}}$ in (2), under assumption that the wavelet is more regular than the process. The following *local singularity strength measure* in 3-D can be defined using wavelets

$$\alpha^i(\mathbf{t}) := \lim_{\mathbf{k}2^{-\mathrm{j}} \to \mathbf{t}} -\tfrac{1}{j}\log_2|d^i_{j,\mathbf{k}}| \tag{3}$$

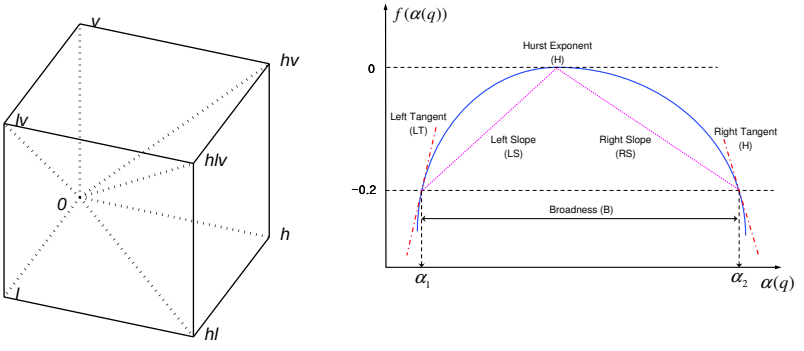


Fig. 2. Seven directions on a cube corresponding to detail level hierarchies in a 3-D wavelet transforms (left); Summary measures (descriptors) from a schematic multifractal spectrum (right)

where $\mathbf{k}2^{-j} \to \mathbf{t}$ means that $\mathbf{t} = (t_1, t_2, t_3) \in [2^{-j}k_1, 2^{-j}(k_1+1)) \times [2^{-j}k_2, 2^{-j}(k_2+1)) \times [2^{-j}k_3, 2^{-j}(k_3+1))$ for $\mathbf{k} = (k_1, k_2, k_3)$ and $j \to \infty$. Smaller $\alpha(\mathbf{t})$ corresponds to larger oscillations in $\mathbf{x}$ and thus to more singularity at time $\mathbf{t}$. The index i in (3) corresponds to one of seven directions in detail spaces of 3-D wavelet transform, horizontal (h), vertical (v) or diagonal (l) up to horizontal, vertical, diagonal (hlv). Typically, a process will possess many different singularity strengths. The frequency (in $\mathbf{t}$) of occurrence of a given singularity strength α is measured by the *3D multifractal spectrum*, defined for each direction i as

$$\begin{aligned} f^i(\alpha) &:= \lim_{\epsilon \to 0} \lim_{j \to \infty} \tfrac{1}{j} \log_2 M_j^i \\ M_j^i &:= 2^{-j} \#\{\mathbf{k} : 2^{-j(\alpha+\epsilon)} \le |d_{j,\mathbf{k}}^i| \le 2^{-j(\alpha+\epsilon)}\}. \end{aligned} \tag{4}$$

For $\mathbf{k} \in \{0, \ldots, 2^j - 1\} \times \{0, \ldots, 2^j - 1\}$, f^i takes values between -1 and 0. Smaller $f^i(\alpha)$ means that "fewer" points $\mathbf{t}$ behave with strength $\alpha(\mathbf{t}) \simeq \alpha$.

The 3-D multifractal spectrum f^i defined in (4) is very hard to calculate. A simpler approach makes use of the theory of large deviations (see Ellis, [8]), where f^i would be interpreted as the rate function of a Large Deviation Principle: f^i measures how frequently (in $\mathbf{k}$) the observed $(-1/j)\log_2 |d_{j,\mathbf{k}}^i|$ deviate from the "expected value" α_0 in scale j. In our 3-D context, it corresponds to studying the scaling behavior of the moments of the wavelets coefficients (2). For every direction i, the *partition function* is defined as follows

$$T^i(q) := \lim_{j \to \infty} (-1/j) \log_2 \mathbf{E} |d_{j,\mathbf{k}}^i|^q. \tag{5}$$

It describes the limiting behavior of qth moment of a typical wavelet coefficient $d_{j,\mathbf{k}}^i$ from the level j and direction i. The *multifractal formalism* posits that the multifractal spectrum can be calculated by taking the Legendre transform of the corresponding log moment generating function (Riedi et al. [19])

$$f^i(\alpha) = f_L^i(\alpha) := \inf_q [q\alpha - T^i(q)]. \tag{6}$$

It can be shown that $f_L^i(\alpha) = q\alpha - T^i(q)$ at $\alpha^i = T'^i(q)$ provided $T''^i(q) < 0$.

3.3 Wavelet-Based Estimator

We discuss in this section wavelet-based estimation of the 3-D multifractal spectrum (4). Given a realization of the 3-D fBm of size $2^J \times 2^J \times 2^J$, and using the stationarity of the wavelet coefficients $\{d^i_{j,(k_1,k_2,k_3)}, i = h, l, v, hl, hv, lv, hlv; j = J_0, \ldots, J-1, k_1, k_2, k_3 = 0, \ldots, 2^j - 1\}$, the sample counterpart of $\mathbf{E}|d^i_{j,\mathbf{k}}|^q$ is

$$\widehat{S}^i_j(q) := \frac{1}{2^{3j}} \left(\sum_{k_1=0}^{2^j-1} \sum_{k_2=0}^{2^j-1} \sum_{k_3=0}^{2^j-1} |d^i_{j,(k_1,k_2,k_3)}|^q \right) \tag{7}$$

for $q > -1$. The partition function can then be estimated as the power-law exponent of the variation of $\hat{S}^i_j(q)$ versus scale 2^{-j}. By linear regression of $\log_2 \hat{S}^i_j(q)$ on j between scales j_1 and j_2 we get

$$\widehat{T}^i(q) := \sum_{j=j_1}^{j_2} a_j \log_2 \widehat{S}^i_j(q), \tag{8}$$

where the regression weights a_j must satisfy the two conditions $\sum_j a_j = 0$ and $\sum_j j a_j = 1$ (Delbeke and Abry [7]). Thus, we can estimate $f^i(\alpha)$ though a local slope of $\widehat{T}^i(q)$ at values

$$\widehat{\alpha}^i(q_l) = [\widehat{T}^i(q_{l+1}) - \widehat{T}^i(q_l)]/q_0, \quad q_l = l q_0$$

as

$$\widehat{f}^i(\alpha^i(q_l)) = q_l \alpha^i(q_l) - \widehat{T}^i(q_l).$$

Multifractal spectra can be found even for monofractal processes; the spectra generated from monofractal processes are ramp-like with a dominant (modal) irregularity corresponding to the theoretical Hurst exponent (see Riedi [17]).

Rather than operating with multifractal spectra as functions (densities), we summarize them by a small number of meaningful descriptors. Each multifractal spectrum (in each direction) can be approximately described by 3 canonical descriptors without loss of the discriminant information, which are (1) Spectral Mode (Hurst exponent, H), (2) left slope (LS) or left tangent (LT) and (3) width spread (Broadness, B) or right slope (RS) or right tangent (RT). A typical multifractal spectrum can be quantitatively described as shown in Figure 3. Understanding the H and LS (or LT) is straightforward. H represents the apex of the spectrum or the Hurst exponent, and LS (or LT) represents the slope of the distribution produced by the collection of Hurst exponents with smaller values of the mode (H). However, broadness (B) is a more intricate descriptor of the multifractal spectrum. Broadness (B) is believed to be more meaningful

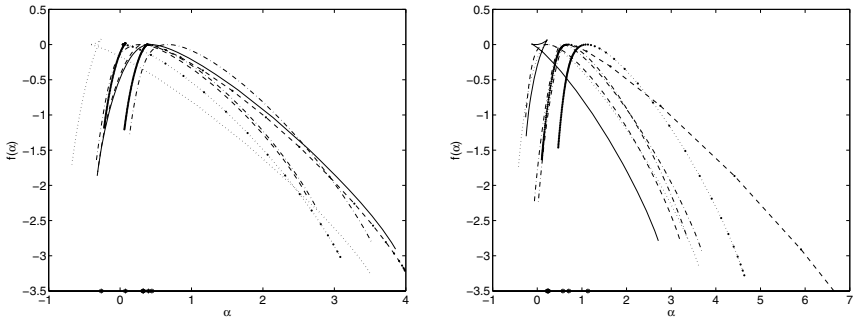


Fig. 3. 3D MFS associated to a 3D fBm with $H = 0.3$ (left) and $H = 0.7$ (right), respectively

than right slope (RS) or right tangent (RT) because it is a compound measure representing the overall nature of the multifractal spectra, taking into account the overall variability among the Hurst exponents. In addition, broadness (B) partially accounts for right slope (RS) or right tangent (RT) in calculation, as the resultant value of B is based on the relative values of RS and LS. Both slopes (or both tangents) can be obtained easily using the interpolation technique, while it is not straightforward to define the broadness (B) automatically. The location of the start and end points of the width spread has been set to the roots α_1 and α_2 which satisfy the equation $f(\alpha)+0.2 = 0$ as in [25]. Figure 3 depicts the MFS of a simulated 3-D fBm with $H = 0.3$ and $H = 0.7$. Notice how the maximum of every $f^i(\alpha)$ is attained close to $\alpha = 0.3$ and $\alpha = 0.7$, and deviations from the exact values can be attributed to discretization or small number of dyadic levels.

4 Application in Analysis of BMRI Images

In this section, we provide an application of the previously defined 3-D wavelet-based multifractal spectrum to the classification of BMRI images. We classify images as benign or malignant, by analyzing the fractal properties of the background of the image. Each image was divided in non-overlapping subimages, each of size $256 \times 256 \times 256$. Each 3-D image contains 104 slices of 256×256 scans that are boundary mirror extended to obtain "wavelet friendly" dimension of 256.

Figure 4 displays 256×256 BMRI slices (cross-sections) from a cancer case and from a control (non-cancerous) subject.

Figure 5 shows particular multifractal descriptors (see also Fig. 2 and its caption) in selected directions for the BMRI data containing two cases and two controls. Fig. 5 (a) displays two selected descriptors, namely H and LS, since they are easily interpretable and appear to distinguish features of cases and controls reasonably well. The descriptor H measures the global irregularity of a scan, while LS describes the deviation from mono-fractality. Images with higher

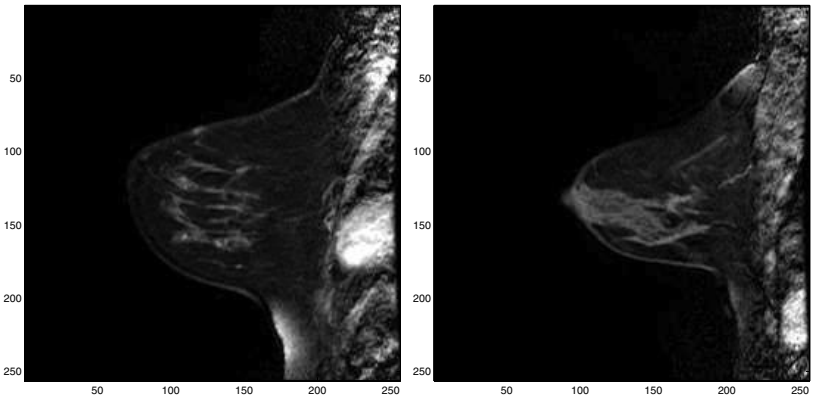
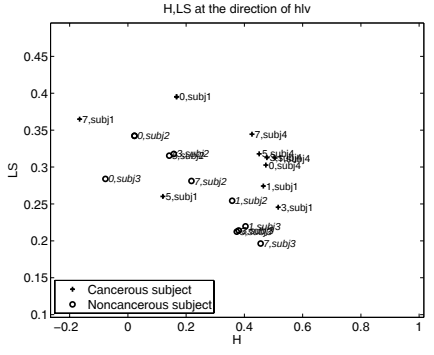
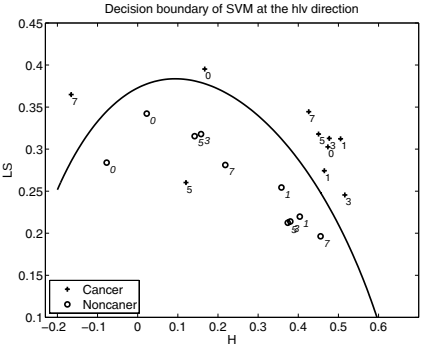


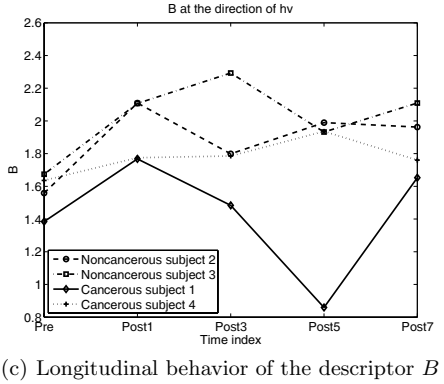
Fig. 4. Examples of case (left) and control (right) BMRI scans, one sagittal slice is shown



(a) Realization of indices (H, LS); the symbols denote a time index, a subject numer, and cancer status.



(b) A decision boundary of SVM (radial basis of width 0.1); the symbols denote a time index and cancer status.



(c) Longitudinal behavior of the descriptor B

Fig. 5. Multifractal descriptors and an associated classifier in selected directions for two cases and two controls. Images (a) and (b) use the main `hlv`-direction, while the third image (c) uses the `hv`-direction.

LS values exhibit greater mono-fractality, and a pure monofractal theoretically has an infinite LS. The direction `hlv`, corresponding to the main diagonal, is selected since the hierarchy of wavelet `hlv`-subspaces contains genuine details. That is, coefficients are obtained by applying high-pass filters in all 3 dimensions, while any other hierarchy utilizes at least one low-pass filter. It is evident that the controls are placed in the region for which $H + LS$ is small – indicating that both irregularity and multifractality for controls are high. This is consistent with a popular belief that increased regularity and monofractality are signs of pathology for many high frequency biometric responses (electrocardiogram (ECG), ocular responses, etc).

Fig. 5 (c) shows the longitudinal behavior of the broadness descriptor B for the cases and controls along the direction `hv`. The `hv` direction combines `h` and `v`, which are sampled along directions within slices and between the slices, respectively. The descriptor B is conceptually linked to H, RS and LS. Namely, spectra with low H and small LS tend to have large B. These typical associations are consistent with our findings depicted in Fig. 5 (a). Even with a small sample size, Figure 5 illustrates the discriminatory power of the multifractal descriptors in BMRI applications. In particular, Fig. 5 (b) shows a non-linear decision boundary generated by a support vector machine (SVM) classifier with radial basis kernel of width parameter 0.1. In our application, the SVM classifier achieved 95% accuracy.

5 Conclusions

In this work, we have shown that the extended three dimensional concept of wavelet-based multifractal spectrum can be utilized in classification of BMRI. This tool, which describes various degrees of irregularity in the measured objects, has been widely utilized in several fields (e.g. physics, meteorology, and medicine), where assessing self-similarity and fractality is critical. Our methodology has provided promising results that are consistent with past research. For example, we observed in our data that normal breast tissue tends to be more irregular (with a smaller Hurst exponent) than tumor affected tissue.

The findings in our study are based upon a small data set, for which the applicability of formal classification algorithms is limited. In future research, involving more data, we plan to build and apply a weak classifier based on scaling of BMRI background, which is a novel concept in cancer screening. We applied the flexible SVM classifier that allows for non-linear classification boundaries, and we will consider other state-of-the art methods in future research. Classification will become more statistically reliable with a large data set that we are in the process of obtaining.

Extremely high classification precision will be challenging to attain with a single classifier, given the high degree of noise in MRI measurements and numerical instability of our algorithms due to limited spatial resolution in the images. However, even moderately accurate classifiers may contribute substantially to breast cancer screening, and these so-called *weak* classifiers in our context utilize

information (BMRI background) that is currently ignored and may combine with other weak classifiers (via boosting) to produce clinically useful tools.

References

1. Alterson, R., Plewes, D.B.: Bilateral symmetry analysis of breast MRI. Phys. Med. Biol. 48, 3431–3443 (2003)
2. Bocchi, L., Coppini, G., Nori, J., Valli, G.: Detection and clustered microcalcifications in mammograms using fractals models and neural networks. Medical Engineering & Physics 26, 303–312 (2004)
3. Buckheit, J., Donoho, D.: Wavelab and reproducible research. Technical report, Stanford University (1995)
4. Chen, C., Daponte, J., Fox, M.: Fractal features analysis and classification in medical imaging. IEEE Transactions on Medical Imaging 8, 133–142 (1989)
5. Derado, G., Bowman, F.D., Patel, R., Newell, M., Vidakovic, B.: Wavelet Image Interpolation (WII): A Wavelet-based Approach to Enhancement of Digital Mammography Images. In: Măndoiu, I., Zelikovsky, A. (eds.) ISBRA 2007. LNCS (LNBI), vol. 4463, pp. 203–214. Springer, Heidelberg (2007)
6. Curpen, B.N., Sickles, E.A., Sollitto, R.A.: The comparative value of mammographic screening for women 40-49 years old versus women 50-59 years old. AJR 164, 1099–1103 (1995)
7. Delbeke, L.: Wavelet based estimators for the Hurst parameter of a self-similar process, PhD Thesis, KU Leuven, Belgium (1998)
8. Ellis, R.: Large deviations for a general class of random vectors. Ann. Prob. 12, 1–12 (1984)
9. Gonçalves, P., Riedi, R., Baraniuk, R.: Simple statistical analysis of wavelet-based multifractal spectrum estimation. In: Proceedings 32nd Asilomar Conference on Signals, Systems and Computers, Pacific Grove, CA (1998)
10. Jemal, A., Siegel, R., Ward, E., Murray, T., Xu, J., Thun, M.J.: Cancer Statistics. CA Cancer J. Clin. 57, 43–66 (2007)
11. Kestener, P., Lina, J.M., Saint-Jean, P., Arneodo, A.: Wavelet-based multifractal formalism to assist in diagnosis in digitized mammograms. Image Anal Stereol. 20, 169–174 (2001)
12. Kuklinski, W.S.: Utilization of fractal image models in medical image processing. Fractals 2, 363–369 (1994)
13. Mainardi, L., Passera, K.M., Lucesoli, A., Vergnaghi, D., Trecate, G., Setti, E., Musumeci, R., Cerutti, S.: A Nonrigid Registration of MR Breast Images Using Complex-valued Wavelet Transform. Journal of Digital Imaging (published online February 28, 2007)
14. Moloney, K.P., Jacko, J.A., Vidakovic, B., Sainfort, F., Leonard, V.K., Shi, B.: Leveraging data complexity: Pupillary behavior of older adults with visual impairment during HCI. ACM Trans. Comput.-Hum. Interact. 13(3), 376–402 (2006)
15. Morales, C.J.: Wavelet-based multifractal spectra estimation: Statistical aspects and applications. Ph.D thesis. Boston University Graduate School of Arts and Sciences (2002)
16. Priebe, C.E., Solka, J.L., Lorey, R.A., Rogers, G.W., Poston, W.L., Kallergi, M., Quian, W., Clarke, L.P., Clark, R.A.: The application of fractal analysis to mammographic tissue classification. Cancer letters 77, 183–189 (1994)

17. Riedi, R.: Multifractal Processes in Theory and Applications of Long-Range Dependence. In: Doukhan, P., Oppenheim, G., Taqqu, M.S. (eds.), pp. 625–716 (in Press, 2002)
18. Riedi, R.H.: An Introduction to multifractals, Tech Report, Dept. Statistics, Rice University (1999)
19. Riedi, R., Crouse, M.S., Ribeiro, V., Baraniuk, R.G.: A multifractal wavelet model with applications to TCP network traffic. IEEE Trans. Info. Theory (special issue on multiscale statistical signal analysis and its applications) (1999)
20. Saslow, D., Boetes, C., Burke, W., Harms, S., Leach, M.O., Lehman, C.D., Morris, E., Pisano, E., Schnall, M., Sener, S., Smith, R.A., Warner, E., Yaffe, M., Andrews, K.S., Russell, C.A.: MD for the American Cancer Society Breast Cancer Advisory Group. American Cancer Society Guidelines for Breast Screening with MRI as an Adjunct to Mammography. CA Cancer J. Clin. 57, 75–89 (2007)
21. Smart, C.R., Hendrick, R.E., Rutledge, J.H., Smith, R.A.: Benefit of mammography screening in women ages 40 to 49 years: current evidence from randomized controlled trials. Cancer 75, 1619–1626 (1995)
22. Vidakovic, B.: Statistical Modeling by Wavelets. Wiley, NY, USA (1999)
23. Yoshida, H., Doi, K., Nishikawa, R.M.L., Giger, M., Schmidt, R.A.: An improved computer-assisted diagnosis scheme using wavelet transform for detecting clustered microcalcifications in digital mammograms. Acad. Radiol. 3, 621–627 (1996)
24. Zheng, B., Qian, W., Clarke, L.P.: Digital mammography: mixed feature neural network with spectral entropy decision for detection of microcalcifications. Med. Img. 15, 589–597 (1996)
25. Shi, B., Vidakovic, B., Katul, G.G., Albertson, J.D.: Assessing the effects of atmospheric stability on the fine structure of surface layer turbulence using local and global multiscale approaches. Physics of Fluids 17 (2005)

Accurate Inverse Consistent Non-rigid Image Registration and Its Application on Automatic Re-contouring

Qingguo Zeng and Yunmei Chen

Department of Mathematics, University of Florida
Little Hall 358, Gainesville, FL 32611-8105
{qingguo, yun}@math.ufl.edu
http://www.math.ufl.edu/~{qingguo,yun}

Abstract. This paper provides a novel algorithm for invertible non-rigid image registration. The proposed model minimizes two energy functionals coupled by a natural inverse consistent constraint. Both of the energy functionals for forward and backward deformation fields consist a smoothness measure of the deformation field, and a similarity measure between the deformed image and the one to be matched. In this proposed model the similarity measure is based on maximum likelihood estimation of the residue image. To enhance algorithm efficiency, the Additive Operator Splitting (AOS) scheme is used in solving the minimization problem. The inverse consistent deformation field can be applied to automatic re-contouring to get an accurate delineation of Regions Of Interest(ROIs). The experimental results on synthetic images and 3D prostate data indicate the effectiveness of the proposed method in inverse consistency and automatic re-contouring.

1 Introduction

Image registration, a very important subject in computer vision and image processing, has been increasingly used in image guided surgery, functional brain mapping, multi-modality fusion etc. The task of image registration is to find a transformation h that relates points in the source image S to their corresponding points in the target image T. This transformation can be either rigid or non-rigid (deformable). Rigid registration is restricted to be a combination of scaling, rotation and translation only, hence, it is not adequate for applications involving large free deformations, for example, image guided radiation therapy on prostate. Deformable image registration allows more freedom at each point, it has been the subject of extensive study in the literature (e.g. [1,20,19,24,14,17,23,10,5]). In most deformable registration models, a smooth and "natural" deformation field h is usually driven by intensity based similarity measures such as Sum of Square Distance(SSD) ([14,16]), Cross Correlation (CC)([6,4]), Mutual Information (MI) ([22,21,6,18]) between the deformed source image $S(h(\mathbf{x}))$ and the target image $T(\mathbf{x})$.

I. Măndoiu, R. Sunderraman, and A. Zelikovsky (Eds.): ISBRA 2008, LNBI 4983, pp. 293–304, 2008.

In certain applications such as imaging guided radiation therapy, it would be better to have a one-to-one and inverse consistent deformation field, while the majority of non-rigid registration methods do not guarantee such property. The inverse consistent means that when the source and target images are switched in the model, the point correspondence between S and T does not change. An inverse inconsistent deformation field can generate large errors in the processes like auto re-contouring([16]), dose calculate ([11,15]) in radiation therapy. A number of work have attempted to make the registration inverse consistent (e.g. [3,12,2]). Here we only discuss two of them which are closely related to our work. In [3], Christensen and Johnson proposed the following coupled minimization problems:

$$\begin{aligned} E(h) &= \underbrace{M(S(h),T) + \lambda R(h)}_{E_1} + \rho \int_\Omega |h - g^{-1}|^2 dx \\ E(g) &= \underbrace{M(S,T(g)) + \lambda R(g)}_{E_2} + \rho \int_\Omega |g - h^{-1}|^2 dx \end{aligned} \tag{1}$$

where $M(\cdot,\cdot)$ is a dissimilarity measure between two images, g is the backward mapping which deforms T such that $T(g)$ is close to S under measure M, and g is expected to be the inverse of the forward mapping h (i.e. $h \circ g = g \circ h = id$, where id is the identity mapping). $R(\cdot)$ is a regularity measure on deformation fields h and g, $\lambda > 0$ and $\rho > 0$ are parameters balances the goodness of alignment, the smoothness of the deformation, and consistence of invertibility. g^{-1} and h^{-1} represent numerical inverses of g and h respectively. In [3], h and g were solved by using gradient descent algorithm.

Since the inverse consistent constraints are accommodated by penalty terms in the energy functionals, solutions h and g by (1) are not exactly inverse to each other. How h and g are closely inverse to each other depends on how large the parameter ρ is, which in practice is hard to choose and needs to be adjusted case by case. Theoretically, h and g are exactly inverse to each other only when $\rho \to +\infty$.

In [12], Leow et al. proposed a different approach. They find h and g by the time marching scheme:

$$h^{(n+1)} = h^{(n)} + dt(\eta_1 + \eta_2), \;\; g^{(n+1)} = g^{(n)} + dt(\xi_1 + \xi_2), \tag{2}$$

where dt is the time step, η_1 and ξ_2 are vector fields representing gradient descent directions of E_1 and E_2 in (1) respectively, i.e.

$$\begin{aligned} \eta_1(x) &= -\nabla_h M(S(h(x)),T(x)) - \lambda \nabla_h R(h(x)) \\ \xi_2(x) &= -\nabla_g M(S(x),T(g(x))) - \lambda \nabla_g R(g(x)). \end{aligned} \tag{3}$$

To make the model inverse consistent, η_1 and ξ_2 are chosen by the following approach.

Suppose $h^{(n)} \circ g^{(n)} = id$ in the nth iteration, then η_2, ξ_1 were determined by taking care of the inverse consistent constraints $h^{(n+1)} \circ g^{(n+1)} = id$, i.e.

$$(h^{(n)} + dt(\eta_1 + \eta_2)) \circ (g^{(n)} + dt(\xi_1 + \xi_2)) = id. \tag{4}$$

Taking the Taylor's expansion of (4) with respect to dt at 0, and collecting up to the first order terms of dt, one gets

$$\eta_2(x) = -D(h(x))\xi_2(h(x)),\ \xi_1(x) = -D(g(x))\eta_1(g(x)) \tag{5}$$

where D is the Jacobian matrix operator. Relations in (5) make the iterations in (2) uni-directional, i.e., updating formula for the forward mapping h does not depend on the backward mapping g, and vise versa.

In this scheme, the driving force for updating h (or g) involves both forward force from E_1 and backward force from E_2, so the scheme aligns two images faster than the models in which the force driving the deformation field depends on E_1 or E_2 only([23]).

However, $h^{(n+1)}$ and $g^{(n+1)}$ by (2-5) are not exactly inverse to each other even $h^{(n)}$ and $g^{(n)}$ are. Since in the derivation of (5), the higher order terms in the Taylor expansion of the left hand side of (4) have been discarded. This generates truncation errors, which are accumulated and exaggerated during iterations. Started with the identity mapping for both h and g, the solutions h and g from (2-5) are not inverse to each other, as we will show in experimental results.

Regarding the dissimilarity measure $M(\cdot,\cdot)$, a conventionally used one for same modality image registration is SSD, which is sensitive to the presence of noise and outliers (e.g. [7,8]). Moreover, the fixed parameter λ in (1) balancing the smoothness of the deformation field and goodness of the alignment is always difficult to select, and affects the robustness of the model to the choice to this weighting parameter. Small λ results an unstable and discontinuous deformation field, while large λ leads to inaccurate result, and may yield a nonphysical deformation field due to unreasonable restrictions. In our proposed model we will replace the SSD dissimilarity measure by a likelihood estimation that is based on the assumption of a Gaussian distribution of the residue image.

The main contribution of this work is on the improvement of inverse consistency, and its application to the radiation therapy, in particular, to get more accurate auto re-contouring. Our basic idea is minimizing E_1 and E_2 in (1) coupled by the inverse consistent constraints defined in the next section. Applications of these inverse consistent deformations on auto re-contouring will be discussed in experimental results.

2 Proposed Method

In this section, we will first introduce a "natural" formulation of inverse consistent constraints, which can be used to correct the truncation errors in (4). Then we will combine this with the dissimilarity measures based on the likelihood of the residue image into our energy functionals to improve the accuracy, robustness and inverse consistent of the deformable image registration.

Let u and v be the forward and backward displacement fields related to deformation h and g by

$$h(\mathbf{x}) = \mathbf{x} + \mathbf{u}(\mathbf{x}),\ \ g(\mathbf{x}) = \mathbf{x} + \mathbf{v}(\mathbf{x}). \tag{6}$$

Then the inverse consistent constraint $h(g(\mathbf{x})) = \mathbf{x}$ can be written in terms of u and v as

$$\mathbf{x} = h(g(\mathbf{x})) = g(\mathbf{x}) + u(g(\mathbf{x})) = \mathbf{x} + v(\mathbf{x}) + u(\mathbf{x} + v(\mathbf{x})). \tag{7}$$

Therefore,

$$v(\mathbf{x}) = -u(\mathbf{x} + v(\mathbf{x})) \tag{8}$$

Similarly, the constraint $g(h(\mathbf{x})) = \mathbf{x}$ can be represented by

$$u(\mathbf{x}) = -v(\mathbf{x} + u(\mathbf{x})) \tag{9}$$

In this work, we will use (8) and (9) as hard constraints in our proposed energy minimization method.

To accommodate certain degree of variability in the image matching, we consider the residue between the deformed source image and target image $S(h(\mathbf{x})) - T(\mathbf{x})$ at each point as an independent random variable with Gaussian distribution of mean zero and a variance σ to be optimized. By the independency assumption, the joint pdf of all these random variables, which is the likelihood of the residual image given parameter σ, becomes

$$\begin{aligned} p(\{S(h(\mathbf{x})) - T(\mathbf{x}), \mathbf{x} \in \Omega\} | T(\mathbf{x}), \sigma) =& \Pi_{\mathbf{x}\in\Omega} p(S(h(\mathbf{x})) - T(\mathbf{x}) | T(\mathbf{x}), \sigma) \\ =& \Pi_{\mathbf{x}\in\Omega} \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{|S(h(\mathbf{x})) - T(\mathbf{x})|^2}{2\sigma^2}}. \end{aligned} \tag{10}$$

Then the negative log-likelihood function is

$$\int_\Omega \frac{|S(h(\mathbf{x})) - T(\mathbf{x})|^2}{2\sigma^2} d\mathbf{x} + |\Omega| \ln \sqrt{2\pi}\sigma \tag{11}$$

Replace $M(S(h), T)$ in E_1 of (1) by (11), and replace $M(S, T(g))$ in E_2 in a similar manner, E_1 and E_2 can be rewritten in terms of u and v as:

$$\begin{aligned} E_1(u) &= \int_\Omega \frac{|S(\mathbf{x} + u(\mathbf{x})) - T(\mathbf{x})|^2}{2{\sigma_1}^2} d\mathbf{x} + |\Omega| \ln \sigma_1 + \lambda R(u(\mathbf{x})) \\ E_2(v) &= \int_\Omega \frac{|S(\mathbf{x}) - T(\mathbf{x} + v(\mathbf{x}))|^2}{2{\sigma_2}^2} d\mathbf{x} + |\Omega| \ln \sigma_2 + \lambda R(v(\mathbf{x})) \end{aligned} \tag{12}$$

After choosing $R(\bullet) = |\bigtriangledown \bullet|^2_{L^2(\Omega)}$ with boundary and initial conditions:

$$\begin{aligned} \frac{\partial u}{\partial \boldsymbol{n}}(x,t) = 0, \ \frac{\partial v}{\partial \boldsymbol{n}}(x,t) = 0, \ \text{ on } \partial\Omega \times R^+ \\ u(x,0) = 0, \ v(x,0) = 0, \ \text{ on } \Omega, \end{aligned} \tag{13}$$

ξ_1 and η_2 in (3) become

$$\begin{aligned} \eta_1(x) &= \frac{T(x) - S(x + u(x))}{\sigma_1^2} \nabla S(x + u(x)) + \lambda \triangle u(x) \\ \xi_2(x) &= \frac{S(x) - T(x + v(x))}{\sigma_2^2} \nabla T(x + v(x)) + \lambda \triangle v(x) \end{aligned} \tag{14}$$

Taking the first variation of the energy functional (12), we get

$$\sigma_1^2 = \frac{\int_\Omega |S(x+u(x)) - T(x)|^2 dx}{|\Omega|},\ \sigma_2^2 = \frac{\int_\Omega |S(x) - T(x+v(x))|^2 dx}{|\Omega|}. \quad (15)$$

By using this dissimilarity measure, the residual image no longer needs to be pointwisely close to zero to make the L^2 norm small. Instead, the new measure only forces the mean of the residue to be zero, and allows the residue having a variance to accommodate certain variability. This is especially good for aligning two images whose intensities are not exactly equal or linearly related, and makes the model more robust to noise and artifacts.

Moreover, the likelihood based approach is less sensitive to the choice of the parameter λ. In SSD models, λ is prefixed, so the balance of the dissimilarity measure and regularity measure does not change during iterations. This makes the selection of λ very difficult and affects registration result. In the proposed model the balancing factor of these two measures is, in fact, $\lambda\sigma^2$ rather than λ alone. Therefore, even λ is prefixed, the weight between these two measures varies at each iteration as the variance updates. As the iterations gradually approach to convergence stage, the residue magnitude becomes smaller, hence, the variance σ reduces, and consequently, the weight on smoothing deformation field versus matching images automatically decreases.

Combining these ideas, we propose a new model to improve the inverse consistency of (1) by using the hard constraints (8) and (9) to replace the penalty terms in (1), and to improve the efficiency of alignment by using the proposed similarity measure. More precisely, we propose to minimize a coupled minimization problem

$$\begin{cases} min_{u\in W^{1,2}(\Omega),\sigma_1}\ E_1(u,\sigma_1) \quad \text{and} \quad min_{v\in W^{1,2}(\Omega),\sigma_2} E_2(v,\sigma_2) \\ \qquad\qquad \text{where } E_1, E_2 \text{ are defined in (12)} \\ \qquad\qquad \textit{subject to } u(x) + v(x+u(x)) = 0\ \forall x \in \Omega \\ \qquad\qquad v(x) + u(x+v(x)) = 0\ \forall x \in \Omega \end{cases} \quad (16)$$

Note if we only minimize $E_1(u)$, the solution u may not be invertible, thus we may not be able to get its inverse through (8). By choosing η_1 and ξ_2 as in (14), the solutions of the constrained coupled minimization problem (16) are obtained via the following algorithm

Algorithm 1

$u(x) = 0; v(x) = 0; iter = 0;$
$corr = correlation(S, T)$
while $iter < maxstep$ and $corr < threshhold$ **do**
  $iter++;$
  $\mathbf{u_{new}} = \mathbf{u} + \mathbf{dt}\eta_1(\mathbf{u})$
  $\mathbf{v} \leftarrow -\mathbf{u_{new}}(\mathbf{x} + \mathbf{v})$
  $\mathbf{v_{new}} = \mathbf{v} + \mathbf{dt}\xi_2(\mathbf{v})$
  $\mathbf{u} \leftarrow -\mathbf{v_{new}}(\mathbf{x} + \mathbf{u_{new}})$

$\mathbf{v} = \mathbf{v_{new}}$
$corr1 = correlation(S(x + u(x)), T(x))$
$corr2 = correlation(S(x), T(x + v(x)))$
$corr = min(corr1, corr2)$
end while

By applying the flow equations alternatively with the exact inverse consistent constraints in Algorithm 1, the forward and backward deformation forces are related via the constraints. In this manner, images to be registered are aligned well, meanwhile, the inverse inconsistency errors are controlled by hard constraints. The performance on image matching and improvement of accuracy on the inverse consistency will be shown in the next section.

To enhance the efficiency of the proposed algorithm, the Additive Operator Splitting scheme([13,9]) is used to speed up our numerical computation. We present our results based on synthetic images, and 3D prostate MRI data.

3 Experiment Results

In this section, we show our experimental results on 2D synthetic images and 3D prostate MRI data, which indicate the improvement of the proposed algorithm in accuracy of inverse consistency and accuracy of auto re-contouring.

Based on (8) and (9), if h and g are inverse to each other, then both of the inverse inconsistent error fields $u(\mathbf{x}) + v(\mathbf{x} + u(\mathbf{x}))$ and $v(\mathbf{x}) + u(\mathbf{x} + v(\mathbf{x}))$ should be 0. Thus, their components and norms should also be 0. This property will be used to compare the inverse inconsistency errors between three methods: Scheme (2), Algorithm 1 and Model (16) without constraints.

The first experiment is aimed to exam and compare the inverse consistency of these three methods on synthetic data. Fig 1(a) and (b) present the source image S and target image T, respectively, with the boundaries of the objects superimposed. Objects have intensity 1.0 in both images, and their background/holes have intensity 0. The three methods are applied separately with the same parameters $dt = .05, \lambda = 5.0$ for 800 iterations, and the corresponding results are shown in Fig 1(c-g). From Fig 1(c), one can see that correlation by all methods converges to about 1.0 similarly. However in Fig 1(d), the means of norms for both inverse inconsistent error fields $u(\mathbf{x}) + v(\mathbf{x} + u(\mathbf{x}))$ and $v(\mathbf{x}) + u(\mathbf{x} + v(\mathbf{x}))$ by both Scheme (2) and non-constrained Model (16) are increased to about 0.4 pixels in average, i.e., the mean value of their inverse inconsistency errors are about 0.4 pixels, while those from Algorithms 1 are maintained in a negligible low level (about 0.01 pixel). We also compare the error field $u(\mathbf{x}) + v(\mathbf{x} + u(\mathbf{x}))$ in Fig 1(e,f,g) by applying results from Scheme (2), Algorithm 1 and Model (16) without constraints respectively on a regular grid mesh. Fig 1(f) shows that the error by Algorithm 1 is 0 almost everywhere(almost no displacement in the regular grid mesh), while (e) and (g) show errors from other two methods are larger, especially in the region corresponding to the boundaries of two holes in S and T.

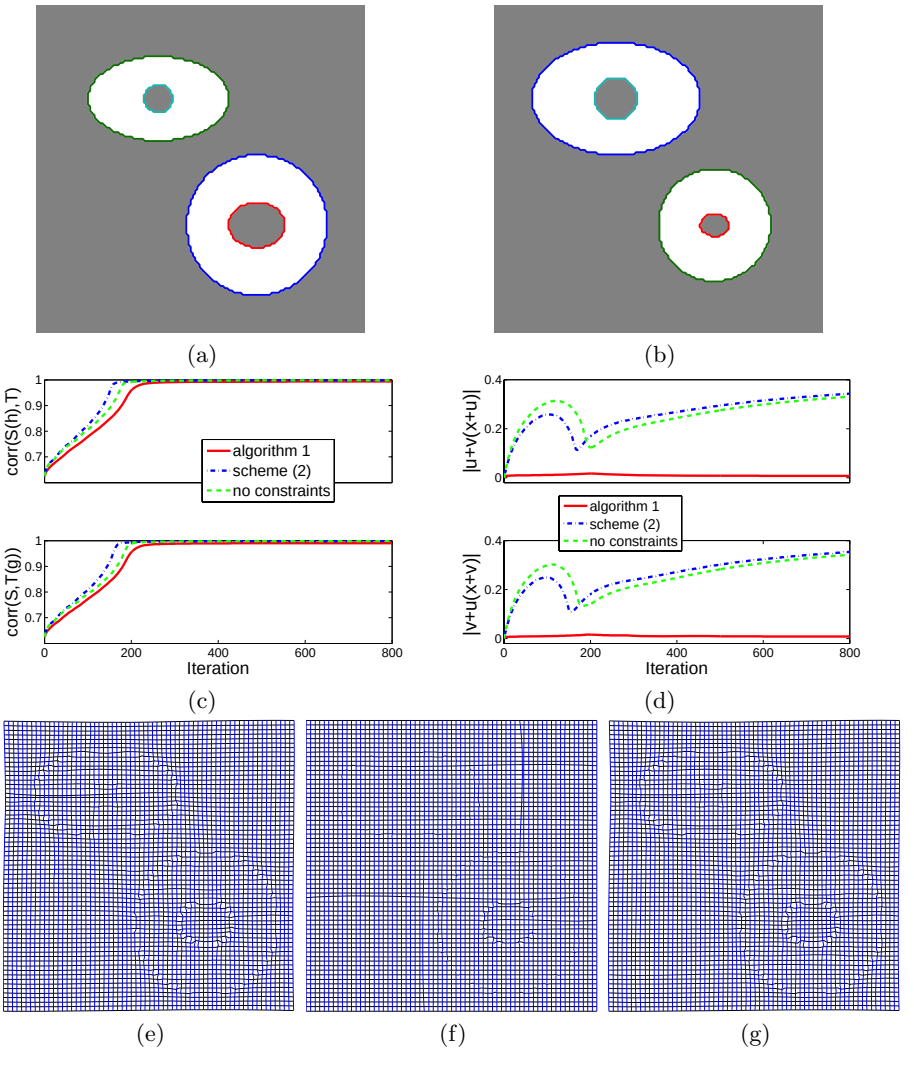


Fig. 1. Experiment on 2D synthetic images. (a) Source with object contours superimposed, (b) target with contours superimposed, (c)correlation $Corr(S(h), T)$ and $Corr(S, T(g))$ during iterations, (d) mean of norms $\|u(x) + v(x + u(x))\|_{L^2(\Omega)}$ and $\|v(x) + u(x + v(x))\|_{L^2(\Omega)}$ during iterations, (e-g) grid representations of inverse inconsistent error field $u(x) + v(x + u(x))$ by Scheme (2), Algorithm 1 with and without constraints respectively.

To quantitatively validate the improved inverse consistency by the proposed algorithm, we compare the maximum and mean values of components and norms of $u(\mathbf{x}) + v(\mathbf{x} + u(\mathbf{x}))$ and $v(\mathbf{x}) + u(\mathbf{x} + v(\mathbf{x}))$ in Table 1, where X_{max}, X_{mean} denote the maximum and mean values of the first component of the error fields

respectively, and Y_{max}, Y_{mean} are those of the second components. The quantitative comparisons are performed in two regions: one is in the image domain Ω, and the other is on the contours which are the boundaries of the objects in the images shown in Fig 1(a,b) respectively. Table 1 shows the proposed algorithm yields much smaller errors in all aspects. Particularly, its mean error is about one fortieth of those from both scheme (2) and non-constrained model (16) in Ω, and about one thirtieth of theirs at the contour regions.

The second experiment is to validate the improvement in accuracy of the proposed algorithm on 3D prostate data, which consists of 100 phases of 2D images

Table 1. Inverse inconsistency error comparison results for synthetic images: the components and norms of inverse inconsistency error fields $u(\mathbf{x}) + v(\mathbf{x} + u(\mathbf{x}))$ and $v(\mathbf{x}) + u(\mathbf{x} + v(\mathbf{x}))$. X_{max}, X_{mean} denote the maximum and mean values of the first component of the error fields respectively, Y_{max}, Y_{mean} denote that for the second component. $\| \bullet \|$ denotes norms of the error fields at each pixel. Ω is the image domain.

inconsistency error $u(\mathbf{x}) + v(\mathbf{x} + u(\mathbf{x}))$ on Ω						
Method	X_{max}	X_{mean}	Y_{max}	Y_{mean}	$\| \bullet \|_{max}$	$\| \bullet \|_{mean}$
Algorithm 1	0.7396	0.0051	0.7876	0.0059	0.7879	0.0085
Scheme (2)	0.9915	0.2390	1.1055	0.2320	1.2272	0.3526
No constraints	0.9916	0.2300	1.0979	0.2255	1.2202	0.3414
inconsistency error $v(\mathbf{x}) + u(\mathbf{x} + v(\mathbf{x}))$ on Ω						
Algorithm 1	0.4870	0.0047	0.5814	0.0053	0.7287	0.0076
Scheme (2)	0.8705	0.2290	1.2224	0.2240	1.2304	0.3398
No Constraints	0.8303	0.2199	1.2670	0.2174	1.2729	0.3284
inconsistency error $u(\mathbf{x}) + v(\mathbf{x} + u(\mathbf{x}))$ on contours in S						
Algorithm 1	0.2810	0.0431	0.2769	0.0502	0.3742	0.0719
Scheme (2)	0.7237	0.1445	0.8047	0.1878	0.8073	0.2577
No Constraints	0.7403	0.1461	0.8374	0.1908	0.8410	0.2612
inconsistency error $v(\mathbf{x}) + u(\mathbf{x} + v(\mathbf{x}))$ on contours in T						
Algorithm 1	0.3833	0.0412	0.4058	0.0484	0.4059	0.0697
Scheme (2)	0.5937	0.1390	0.9308	0.1661	0.9715	0.2361
No Constraints	0.6180	0.1410	0.9735	0.1695	1.0171	0.2402

Table 2. Inverse inconsistency error comparisons for the $1st$ and $21st$ phases of a 3D prostate MRI data on regions Ω and all the contours in the $1st$ phase S

Method	X_{max}	X_{mean}	Y_{max}	Y_{mean}	$\| \bullet \|_{max}$	$\| \bullet \|_{mean}$
inconsistency error field $u(\mathbf{x}) + v(\mathbf{x} + u(\mathbf{x}))$ on Ω						
Algorithm 1	0.5155	0.0105	0.4010	0.0082	0.5549	0.0146
Scheme (2)	0.4936	0.0457	0.6750	0.0793	0.7154	0.1020
inconsistency error field $v(\mathbf{x}) + u(\mathbf{x} + v(\mathbf{x}))$ on Ω						
Algorithm 1	0.5937	0.0121	0.4766	0.0087	0.6230	0.0164
Scheme (2)	1.6315	0.0583	1.5860	0.0951	1.9917	0.1231
inconsistency error $u(\mathbf{x}) + v(\mathbf{x} + u(\mathbf{x}))$ on contours in S						
Algorithm 1	1.6165	0.0524	1.9630	0.0459	2.3577	0.0770
Scheme (2)	4.7525	0.1036	6.2510	0.1292	6.3180	0.1843

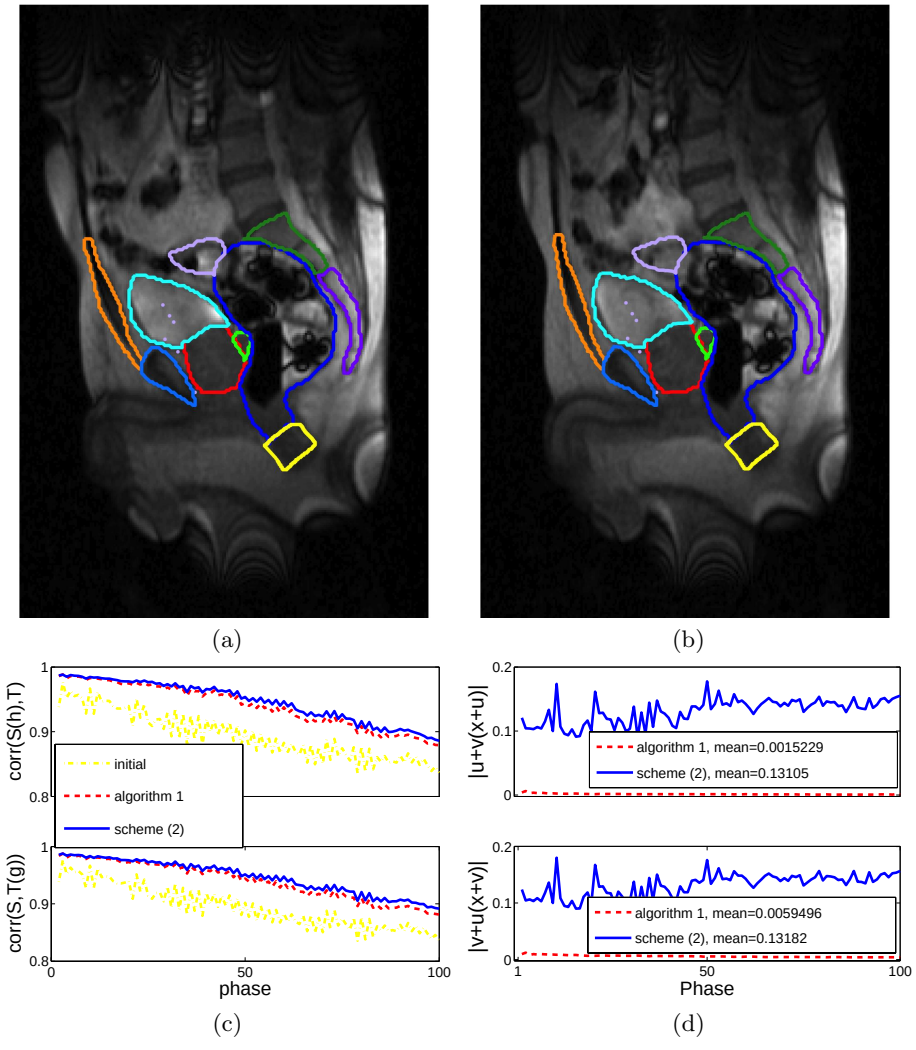


Fig. 2. Experiment on 3D prostate MRI data with 100 2D phases. (a) Source image(the $1st$ phase) with contours of ROIs superimposed, (b) the $21st$ phase with automatic re-contouring results by Algorithm 1 superimposed, (c) plots of correlation $Corr(S(h), T)$, and $Corr(S, T(g))$ between the $1st$ phase and each of other 99 phases, (d) plots of norms $||u(x) + v(x + u(x))||$ and $||v(x) + u(x + v(x))||$ for deformations between the $1st$ phase and each of other 99 phases, (c,d) are based on parameters $\lambda = 20, dt = .1$ for 200 iterations.

focusing on prostate area, where ROIs have large internal motions. The source volume S is the first phase, and the boundaries of ROIs in S are delineated by contours and superimposed in Fig 2(a), and the other 99 phases are targets, and

methods Scheme (2) and Algorithm 1 are applied to find deformations between the $1st$ phase and each of the other 99 phases. The auto re-contouring, that register the contours in S into the other 99 phases, is achieved by applying the deformations on these contours. For demonstration purpose, one of the target image T, the $21st$ phase, is shown in Fig 2(b), and the auto re-contouring result by Algorithm 1 is superimposed on it.

Comparison on the convergence and inverse inconsistency by the first two methods are shown in Fig 2(c,d). Fig 2(c) compares CC between deformed images and target images by these two models. Both Scheme (2) and Algorithm 1 improve the initial CC between S and each of other 99 phases to a similar level. However, from Fig 2(d) we can observe that the norm of inverse inconsistency error by scheme (2) is much higher in average than that of Algorithm 1.

The quantitative comparison on inverse inconsistency errors between the $1st$ phase and the 21st phase for this experiment is listed in Table 2 for demonstration. Beside comparing the error fields on Ω, we also evaluate error $v(x)+u(x+v)$ at points of all given contours on S. By comparing the corresponding components and norms of the inverse inconsistency error fields in Table 2, we find that errors generated by proposed algorithm is much lower than that by scheme (2), this indicates that the point correspondence and automatic re-contouring results are more accurate.

4 Conclusion

In this work, we proposed a coupled energy minimization method with inverse consistent constraints for deformable image registration. The proposed model controls the inverse inconsistency errors in a negligibly low level, therefore, it provides a better correspondence for the ROIs in source and target images. This makes the auto re-contouring results with data involved in the course of radiation therapy much more accurate.

The dissimilarity measure used in this work is the negative log-likelihood of the residual image between the deformed source and target. This dissimilarity measure is able to accommodate certain variability in the matching. Hence, the model is more robust to noise than SSD, moreover, it is less sensitive to the choice of the parameter that balances the smoothness of the deformation field and goodness of matching.

Acknowledgments

Authors would like to thank Dr. Jim Dempsey and Dr. Anneyuko I. Saito from Department of Radiation Oncology at University of Florida and Viewray Inc. for providing the prostate data. The work was partially supported by NIH R01NS052831-01A1.

References

1. Bajcsy, R., Kovacic, S.: Multiresolution elastic matching. Computer Vision, Graphics, and Image Processing 46(1), 1–21 (1989)
2. Beg, M.F., Khan, A.: Symmetric data attachment terms for large deformation image registration. MedImg 26(9), 1179–1189 (2007)
3. Christensen, G., Johnson, H.: Consistent image registration. IEEE Transactions on Medical Imaging 20, 568–582 (2001)
4. Collins, D.L., Evans, A.C.: Animal: Validation and applications of nonlinear registration-based segmentation. IJPRAI 11(8), 1271–1294 (1997)
5. Schreibmann, E., Xing, L.: Narrow band deformable registration of prostate magnetic resonance imaging, magnetic resonance spectroscopic imaging, and computed tomography studies. Int. J. Radiat. Oncol. Biol. Phys. 62, 595–605 (2005)
6. Hermosillo, G., Hotel, C.C., Faugeras, O.: Variational methods for multimodal image matching. Int. J. Computer Vision 50(3), 329–343 (2002)
7. Hill, D., Batchelor, P., Holden, M., Hawkes, D.: Topical review: Medical image registration. Physics in Medicine and Biology 46, 1–45 (2001)
8. Jian, B., Vemuri, B., Marroquín, J.: Robust nonrigid multimodal image registration using local frequency maps. In: 19th International Conference on Information Processing in Medical Imaging, pp. 504–515 (2005)
9. Weickert, J., Romeny, B., Viergever, M.: Efficient and reliable schemes for nonlinear diffusion filtering. IEEE Trans. on Img. Proc. 7(3), 398–410 (1998)
10. Brock, K.K., Sharpe, M.B., Dawson, L.A., Kim, S.M., Jaffray, D.A.: Accuracy of finite element model-based multi-organ deformable image registration. Med. Phys. 32, 1647–1659 (2005)
11. Keall, P.J., Siebers, J.V., Joshi, S., Mohan, R.: Monte carlo as a four-dimensional radiotherapy treatment-planning tool to account for respiratory motion. Phys. Med. Biol. 49, 3639–3648 (2004)
12. Leow, A.D., Huang, S.C., Geng, A., Becker, J., Davis, S., Toga, A., Thompson, P.: Inverse consistent mapping in 3d deformable image registration: Its construction and statistical properties. In: Christensen, G.E., Sonka, M. (eds.) IPMI 2005. LNCS, vol. 3565, pp. 493–503. Springer, Heidelberg (2005)
13. Lu, T., Neittaanmki, P., Tai, X.-C.: A parallel splitting-up method for partial differential equations and its application to navier-stokes equations. RAIRO Math. Model. and Numer. Anal. 26(6), 673–708 (1992)
14. Lu, W., Chen, M., Olivera, G., Ruchala, K., Mackie, T.: Fast free-form deformable registration via calculus of variations. Physics in Medicine and Biology 49, 3067–3087 (2004)
15. Lu, W., Olivera, G., Mackie, T.R.: Motion-encoded dose calculation through fluence/sinogram modification. Med. Phys. 32, 118–127 (2005)
16. Lu, W., Olivera, G.H., Chen, Q., Chen, M., Ruchala, K.: Automatic re-contouring in 4d radiotherapy. Physics in Medicine and Biology 51, 1077–1099 (2006)
17. Coselmon, M.M., Balter, J.M., McShan, D.L., Kessler, M.L.: Mutual information based ct registration of the lung at exhale and inhale breathing states using thin-plate splines. Med. Phys. 31, 2942–2948 (2004)
18. Maes, F., Collignon, A., Vandermeulen, D., Marchal, G., Suetens, P.: Multimodality image registration maximization of mutual information. In: MMBIA 1996: Proceedings of the 1996 Workshop on Mathematical Methods in Biomedical Image Analysis (MMBIA 1996), Washington, DC, USA, p. 14. IEEE Computer Society Press, Los Alamitos (1996)

19. Rogelj, P., Kovacic, S.: Similarity measures for non-rigid registration. In: Proc. SPIE, vol. 4322, pp. 569–578 (2001)
20. Thirion, J.: Image matching as a diffusion process: an analogy with maxwell's demons. Medical image analysis 2(3), 243–260 (1998)
21. Viola, P.A., Wells III, W.M.: Alignment by maximization of mutual information. In: ICCV, pp. 16–23 (1995)
22. Pluim, J.P.W., Maintz, J.B.A., Viergever, M.A.: Mutual-information-based registration of medical images: A survey. IEEE Trans. Med. Imaging 22, 986–1004 (2003)
23. Wang, H., Dong, L., O'Daniel, J., Mohan, R., Garden, A., Ang, K., Kuban, D., Bonnen, M., Chang, J., Cheung, R.: Validation of an accelerated 'demons' algorithm for deformable image registration in radiation therapy. Phys. Med. Biol. 50(12), 2887–2905 (2005)
24. Zitova, B., Flusser, J.: Image registration methods: A survey. Image Vis. comput. 21, 977–1000 (2003)

GlycoBrowser: A Tool for Contextual Visualization of Biological Data and Pathways Using Ontologies*

Matthew Eavenson[1], Maciej Janik[1], Shravya Nimmagadda[1],
John A. Miller[1], Krys J. Kochut[1], and William S. York[2]

[1] Computer Science Department
[2] Complex Carbohydrate Research Center
University of Georgia
Athens, GA 30602
{durandal, mjanik, shravyan}@uga.edu,
{jam, kochut}@cs.uga.edu,
will@ccrc.uga.edu

Abstract. This paper introduces GlycoBrowser, a dynamic biological pathway exploration and visualization tool. Its capabilities include incremental navigation of pathways, as well as overlaying relevant experimental data about glycans and reactions within a pathway. The use of ontologies not only allows dynamic construction of pathways, but also facilitates more accurate validation of the information retrieved. Pathway exploration is initiated by means of an ontologically driven glycan structure building tool, which facilitates glycan structure construction and searching with minimal user error. Because of the complex nature of glycan structures and the difficulty involved in interpreting the associated data, GlycoBrowser is tailored especially to suit the needs of glycobiologists.

Keywords: GlycoBrowser, pathway browsing, metabolic pathway exploration, glycomics, transcriptomics, ontology, semantic web.

1 Introduction

The amount of data and knowledge stored in Web-accessible databases and ontologies is growing explosively. Making this information widely available in an intuitive form is becoming almost as important as creating this information in the first place. Traditionally, Web 1.0 technology, relying on servlets and Java Server Pages (JSP) to interface with relational databases, has been the preferred way to develop such online resources [3]. The approach that we are taking utilizes Web 2.0 as well as Semantic Web technology to develop a more agile means for querying and browsing biological information at a higher level than most on-line resources. The key component of our approach is the pathway exploration and visualization tool, GlycoBrowser.

GlycoBrowser is multi-faceted, allowing flexible visualization of both glycomic pathways and construction of glycan molecules. It leverages the capabilities of

* GlycoBrowser was developed as part of the Integrated Technology Resource for Biomedical Glycomics (P41 RR18502-02), funded by the National Institutes of Health - National Center for Research Resources.

I. Măndoiu, R. Sunderraman, and A. Zelikovsky (Eds.): ISBRA 2008, LNBI 4983, pp. 305–316, 2008.

GlycoVault [18], along with several Web 2.0 and Semantic Web technologies (including RDF [14], OWL [16], SPARQL [20], AJAX, Javascript, and BRAHMS [11]) to represent structural information in a way that is intuitive for glycobiologists and to overlay this information with experimental data (glycomics, proteomics, and transcriptomics analyses). GlycoBrowser provides a graphical display of glycan biosynthetic pathways and associated experimental data. Glycans are rendered "on the fly" using the standard representation endorsed by the Consortium for Functional Glycomics (CFG, http://www.functionalglycomics.org), extended to include partonomy relationships. Relevant experimental data (such as the abundance of a particular glycan in a biological sample) can be shown in associated histograms.

This paper discusses the motivation, design and implementation of GlycoBrowser and is organized as follows. Section 2 considers why more agile, flexible and high level approaches that feature visualization provide a better means of accessing biological information. In section 3, we narrow our focus and review the evolution of biochemical pathway visualization tools and where GlycoBrowser fits in. An important aspect of our approach is that we use knowledge encoded in ontologies, those available on the Web as well as ones we are developing, as discussed in section 4. Section 5 presents the design and implementation of our pathway browser and its associated structure builder. Conclusions and future work are given in section 6.

2 Motivation

A major challenge of modern biological science is to analyze and interpret the huge amount of data that is routinely collected by high-throughput techniques, as in proteomics or glycomics analysis. Due to the complexity of biological systems, it is not sufficient to simply facilitate access to specific data in these large sets; it is necessary to present these data in the context of what is known about the biological system being studied. One approach to address this requirement is to present data (such as proteomics or transcriptomics data) in the context of a metabolic or signaling pathway in a process called “data overlaying” [23]. In the domain of glycobiology (the study of complex carbohydrates and their functional roles in living organisms), such a contextual illustration of data is complicated by the fact that the pathways leading to glycan biosynthesis are extremely complex, and each of the molecules in the pathway is a complex aggregation of smaller “glycosyl residues”. Therefore, graphical representation of the pathway requires a robust internal representation of the structures along with algorithms for rendering them in a format that can be intuitively interpreted “at a glance” by the scientist.

We have developed an ontology (GlycO) that embodies knowledge regarding the structures, biosynthesis, and biological functions of complex glycans [22]. GlycO thus addresses the challenge of providing a robust internal representation of glycan structures and the metabolic pathways leading to their synthesis. In addition, GlycO comprises a repository of knowledge regarding other aspects of glycobiology, including structural and functional relationships between different glycans. GlycoBrowser allows this knowledge to be accessed and used as a context for the graphical representation of experimental data. We initially focus on two types of data: (1) qRT-PCR data, revealing the abundances of mRNA transcripts for genes involved in glycan biosynthesis and (2) glycomics data, revealing the identities and abundances of specific glycans in a biological sample. Ultimately, GlycoBrowser will not only provide a

means of overlaying data on metabolic pathways, it will provide an entry point into the diverse types of knowledge embedded in GlycO and other ontologies.

The following example illustrates the need for a tool with the capabilities of GlycoBrowser. We are interested in the relationships between the differentiation of stem cells (to form more specialized cells) and the expression of specific glycans on their surfaces. Our collaborators in the Integrated Technology Resource for Biomedical Glycomics (http://glycomics.ccrc.uga.edu) have developed and implemented powerful methods for obtaining transcriptomic and glycomic data for this experimental system. The transcriptomic data reveals the expression levels of specific enzymes involved in the metabolic pathways leading to glycan biosynthesis and the glycomics data reveals the amounts of specific glycans that are generated by these complex pathways. Manual analysis of this data indicated that specific glycans (sialylated biantennary N-glycans) are more abundant on the surfaces of differentiated cells than on undifferentiated stem cells. Moreover, some of the enzymes (sialyl transferases, which catalyze the addition of a sialic acid residue to the nascent glycan) required for the biosynthesis of these glycans are also upregulated during differentiation. The question thus arises, "Are the changes in the abundances of these glycans due solely to an increase in sialyl transferase expression, or do changes in the expression of other enzymes contribute significantly to this effect?" In order to answer this question, a glycobiologist might draw all of the relevant pathways and overlay the relevant transcriptomic and glycomic data at each step of the pathway. However, rendering this pathway is not trivial, as each of the molecules along the pathway is a complex glycan, and the pathway is highly branched. It would be difficult to find an appropriate entry point into the pathway (for drawing) and it would also be difficult to select the appropriate branch(es) of the pathway that lead to the glycans of interest.

Our initial implementation of GlycoBrowser addresses these challenges in several ways. It provides a graphical interface that allows an entry point (a specific glycan) to be selected from a large collection of structures in the knowledge base. Glycans are complex, branched molecules. Therefore, it is much more difficult to select a glycan from a collection of glycans than to select a protein sequence from a collection of protein sequences. It would be highly impractical to expect scientists to memorize the accession numbers of thousands of glycan structures, and thus it is necessary to provide a graphical tool for selecting specific glycans. Thus, a tool that enables a search for specific glycans based on their structural features, but disallows structures not found in nature would be quite useful, as it would eliminate time wasted searching for physically impossible combinations of structural features.

GlycoBrowser provides such a tool, and allows the user to define an entry point into the metabolic pathway. Complex, branched metabolic pathways are rendered using an accepted graphical representation of the glycan structures. GlycoBrowser also overlays specific transcriptomic and glycomic data at each step along the pathway. Analysis of our transcriptomic and glycomic data using GlycoBrowser demonstrated that increased expression of sialyl transferases in not the only factor leading to the increased abundances of sialylated biantennary N-glycans in differentiated cells. The abundances of precursor glycans (which are substrates for the sialyl transferases) are also elevated, indicating that other steps in the pathway leading to the sialylated biantennary N-glycans are modified in the differentiated cells.

3 Background

Dynamic molecule and pathway construction is not a new approach. Multiple research areas impose different requirements for visual representation of pathways. As a result, there exist many tools and browsers for visualizing biological structures or pathways that can be enhanced with experimental data and other information. Among the most popular are KEGG [19], WikiPathways [13], MetaCyc [12], PathwayAssist [17], GenMAPP [7], and Cytoscape [24]. Although all these tools are designed to be comprehensive, each emphasizes different aspects of data presentation.

KEGG and WikiPathways offer static pathways visualization, pathway interconnectivity and also allow focus on many levels of granularity such as meta-pathway, pathway fragment, or details of a single compound. They serve as referral sites and repositories of known pathways. Dynamic pathway browsing is offered by MetaCyc, PathwayAssist, GenMAPP, and Cytoscape. These systems allow a user to query for pathway fragments. Their search capabilities may include finding pathways between specific molecules, gene regulators or specific reactions based on given properties. They use a database to find relevant information for visualization and offer multiple layouts and presentation schemes to present focused information.

Information overlay is another important aspect of pathway visualization. Dynamic browsers support enhancing pathway elements with accompanying experimental data or related information from biological databases and special ontologies, such as the Gene Ontology (GO) [10]. Information overlay allows scientists to easily see results of their experiments in proper biological context.

GlycoBrowser is a dynamic visualization tool created specifically to handle complex glycomics data. It supports search for glycans based on their structural features, which are selected using a graphical user interface. Contrary to unrestricted graphical editing in Glycan Builder [6], our system uses an ontology to guide the building process and restricts the user to creating only structures that fit into one of the canonical trees that represent all glycan structures in the ontology [22, 25]. A glycan located using this process may be used as a starting point for further pathway exploration which allows a user to dynamically traverse interesting pathways. Displayed glycans and reactions are then overlaid with associated experimental data. This guided approach to molecule and pathway construction minimizes user error, while also allowing a biologist to explore data, rather than simply sifting through it.

4 Underlying Ontological Representations

As mentioned in the introduction, more and more ontologies are becoming available on the Web. Along with relational databases, they make up the principal means of providing structured information on the Web. Making such information available to end users in meaningful and convenient ways has lead to massive software development over the last decade. In particular in biosciences, many well developed ontologies are now available on the Web. For example, at the Open Biomedical Ontologies (OBO) Foundry [2], there are now over sixty ontologies registered. Finally, an important focus of our glycomics project is the study of glycans and the effect that enzymes

have on their biosynthesis. We have therefore built and are in the process of populating the following two ontologies: GlycO and EnzyO [22, 25]. These ontologies were developed with Protege using the Web Ontology Language (OWL) [16]. For efficiency of pathway browsing, export to Resource Description Framework (RDF) [14] is also provided.

The Glycomics Ontology (GlycO) [25] has been preloaded with the residues that make up glycans. Then to load any glycan into the knowledge-base, one simply needs to add this new instance with links to the existing residues. The canonical approach reduces redundancy and increases the reliability of the information stored, since we at least know that the preloaded residues are correct. In addition to this information and knowledge about structures, GlycO also links structures to reactions that can make them. The population of GlycO is facilitated by GlydeII [26] which is an XML based data interchange standard for glycan structures. GlydeII facilitates validation and incorporation of new glycan structure instances by comparison to knowledge stored in GlycO. The Enzyme Ontology (EnzyO) keeps track of enzymes that catalyze the reactions which produce the glycan structures. The identities and abundance levels of glycans and enzymes are the essential components in the biochemical pathways. The ontology keeps track of basic information about enzymes (e.g., their Enzyme Commission (EC) number, their protein structure) as well as associations (e.g., with the gene that codes for it and the reactions it participates in).

5 GlycoBrowser

GlycoBrowser consists of two submodules, the Canonical Structure Builder and Pathway Browser, and utilizes three related subsystems, the GlycoVault [18] repository, SPARQL Server, and Image Server. We will describe each of these in detail, beginning with the lower level components (GlycoVault, SPARQL Server, and Image Server), and concluding with the Glyco Browser functionality and implementation.

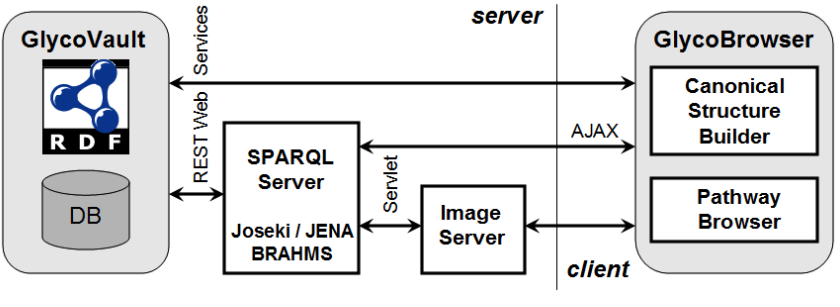


Fig. 1. System architecture

5.1 GlycoVault

GlycoVault provides a means of storing and retrieving data to support glycomics research at the Complex Carbohydrates Research Center (CCRC) at the University of

Georgia. These data include quantitative Real-Time Polymerase Chain Reaction (qRT-PCR) data as well as basic glycomics data, such as biologically relevant parameters and various types of experimental data, along with the explicit and implicit knowledge required to analyze and interpret these data. GlycoVault consists of databases, ontologies (including GlycO and EnzyO), and data files in various formats. These datasets and ontologies are accessed by the Pathway Browser via a comprehensive, yet easy to use Application Programming Interface (API). The API facilitates the development of methods for querying the knowledge and exporting the results in formats (such as XML, RDF or OWL) that can be readily incorporated by external applications. GlycoVault provides easy to use query interfaces, including support for the SQL and SPARQL query languages.

5.2 SPARQL Server

The SPARQL Server acts as an ontology storage and querying facility for the Pathway Browser. It provides access to any ontological knowledge available, whether it be the next possible branch at each step along a pathway, or any additional metadata about reactions or molecules. SPARQL was the obvious query language of choice, since we are taking an ontology-based approach and require a query language that works with RDF (or OWL). Currently, we use Joseki [1], a Jena [5] based HTTP engine that supports the SPARQL Protocol and query language. Joseki allows responses to be encoded in either XML or JavaScript Object Notation (JSON) [21]. We have chosen JSON because it meshes seamlessly with the Javascript user interface. However, as an added benefit, JSON, being a lightweight data-interchange format, is much less verbose than an XML encoding, meaning less data to transfer over the web and less bandwidth consumed. In the near future, we intend to migrate to a high-performance BRAHMS based server based on the SPARQLeR [15] extension to SPARQL. This will allow us to retrieve entire pathway fragments from the ontology, allowing fragments to be constructed in their entirety within the Pathway Browser instead of incrementally as described later.

5.3 Image Server

The Image Server dynamically constructs cartoonist [9] representations of glycan molecules, based on the underlying ontological structure. The cartoonist representation has been widely accepted and endorsed by the Consortium for Functional Glycomics. Patterns present in a cartoonist representation (as depicted in Figures 2, 3, and 4) are far more readily recognizable at-a-glance than the name of a glycan molecule, or its textual representation, making it the obvious approach for implementing a pathway browser whose purpose is to be intuitive. However, the complexity of dynamically generating cartoonist models of glycans, in tandem with the desire for modular design, necessitated the creation of a separate, dedicated image drawing subsystem. We felt it was preferable to shift drawing to a dedicated module both for speed and for making the client more lightweight. The benefits of having a fast, dynamic, and modular image drawing subsystem connected directly to the GlycO ontology are readily apparent, as any changes to the underlying structures in the ontology will not require the generation of new static images. The resulting images are used in

both the Canonical Structure Builder and Pathway Browser, and can also be utilized in any future extensions.

The image construction algorithm itself is implemented in C++, and makes use of the BRAHMS API (one of the fastest currently available RDF stores) for storage and querying of RDF data, as well as FastCGI [4], which facilitates efficient querying over the web. We utilize the GraphViz API [8] to automatically draw glycan molecules. However, we have found that carefully tweaking the GraphViz input can provide more suitable layouts for glycan molecules. All internal molecule representations are generated dynamically by recursively traversing the structure of the GlycO ontology, piecing together the component residues of a glycan molecule. The resulting graph structure is then passed to GraphViz which renders the collected nodes and edges into a cartoonist representation of a glycan. We ultimately chose GraphViz because of its easy to use API, choice of image formats (SVG, PNG, JPEG, GIF), and straightforward image customization options. Our image format of choice is PNG, owing to its non-proprietary format and improved compression characteristics over GIF.

5.4 Canonical Structure Builder

The Canonical Structure Builder serves as an entry point to pathway visualization, as well as a convenient search tool to look for glycan structures. The user may graphically construct a glycan molecule using component residues. However, in our novel approach, the construction is guided by the underlying ontological structure, thus reducing the amount of possible user error. The structure can then be matched against glycans currently represented in the ontology, either finding larger glycans which contain it, or the exact glycan itself.

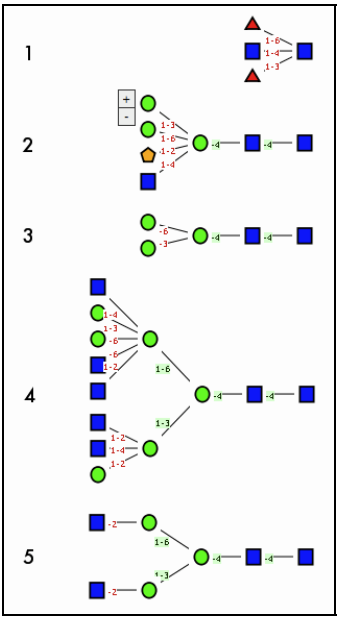


Fig. 2. Structure builder scenario

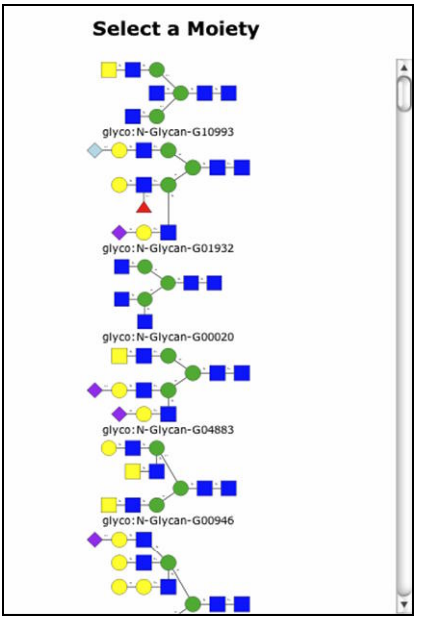


Fig. 3. All matching glycans

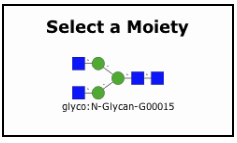


Fig. 4. Exact matching glycan

As a sample scenario (Figure 2), the user is first presented with a list of available root residues of the glyco tree to choose from. Once a root is chosen, the residue is rendered into the main viewport, with any connected residues branching off from it to the left (step 1). The user can then construct a glycan representation by expanding or collapsing residues based on the glyco tree until a desired configuration is achieved (steps 2-4). Residues are expanded or collapsed through the use of a simple "plus/minus" toggle that appears when mousing over the residue, as depicted in step 2. Once the final molecule is constructed (step 5), the user can then search for the structure in the knowledge-base. The "Match Glycans" button is used to search for all glycans which contain the given structure, with the results arranged in a list as presented in Figure 3. However, if a user wants to search for only that particular configuration, then "Exact Match Glycans" can be used, as shown in figure 4. Selecting a glycan from the results initiates pathway exploration starting from the selected glycan.

5.5 Pathway Browser

The Pathway Browser allows exploration of a biological pathway beginning from a user-selected glycan. In keeping with our desire for maximum usability, the interface layout has been kept as simple and intuitive as possible. The pathway is created dynamically, with the data that forms the pathway structure itself coming from the SPARQL Server, and any related experimental data coming from the GlycoVault.

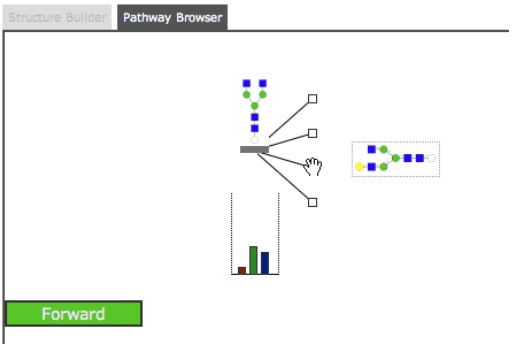


Fig. 5. Start of pathway exploration

The pathway is rendered as a series of nodes and edges. Molecules are rendered by the Image Server (discussed previously), which makes use of GlycO, the same ontology used for constructing the pathway. Reactions which have not yet been expanded

are rendered as small squares, while already expanded reactions are rendered as arrows between molecules. Positioning the mouse pointer over a reaction that has not been expanded previews the molecules resulting from the reaction beside it, as depicted in Fig. 5. Exploring a pathway consists of expanding one potential branch point at a time. Potential branch points within a pathway consist of the molecules or reactions themselves, as a molecule can be used as a substrate in multiple reactions, and a reaction can have multiple products.

Related experimental data is rendered into small bar graphs located directly beneath molecules and reactions. The data under molecules represent glycan abundances, while the data under reactions represent transcriptome expression levels (corresponding to enzyme abundance levels). If experimental data is unavailable, then the corresponding graph is left empty. However, if there is more than one set of data for a particular node, the graph will present controls allowing the user to cycle through them.

A sample pathway fragment illustrating the described features is depicted in Fig. 6, starting from the constructed glycan, as described in the previous section. This figure illustrates the answer to the aforementioned question, "Are the changes in the abundances of these glycans due solely to an increase in sialyl transferase expression, or do changes in the expression of other enzymes contribute significantly to this effect?" In fact, a glycobiologist will have the relevant pathway with associated overlaid transcriptomic and glycomic data at each step of the pathway.

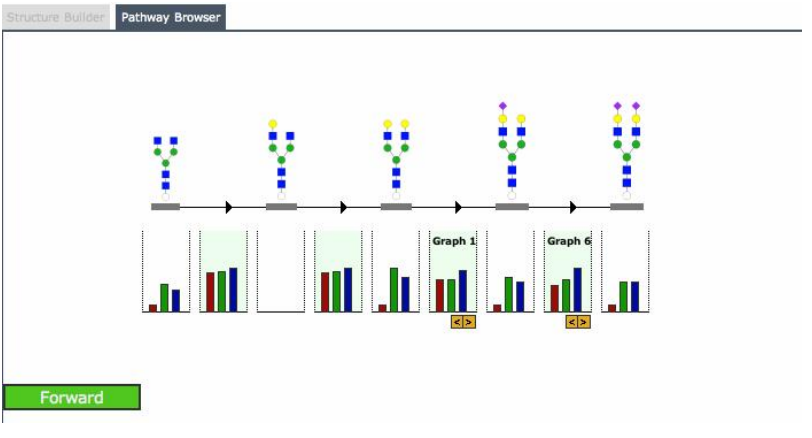


Fig. 6. Pathway fragment with corresponding graphs

Additionally, the pathway traversal direction can be toggled by clicking on the Forward/Backward button, with the currently displayed label dictating which way the pathway will be traversed. A glycobiologist may navigate forward or backward on a pathway to pinpoint where abundance levels significantly change. Backtracking to any point in the pathway is also possible by clicking on an already expanded node. This allows revisiting prior points along the pathway, as well as exploring other branches.

5.6 Implementation Choices

We are utilizing both Web 2.0 technology as well as Semantic Web technology to provide information at a higher level than offered by most on-line resources. The user interfaces for both the Canonical Structure Builder and the Pathway Browser are implemented using Javascript. We chose Javascript because a web accessible pathway browsing tool will be most useful for glycobiologists, as it can be used from almost any location, and provides platform independence. The decision to favor Javascript as opposed to a Java applet was also made to keep the interface lightweight, as downloading an applet is time and bandwidth consuming as well as problematic on certain web browsers.

Moreover, the decision to use Javascript allows us to utilize the AJAX framework to incrementally download new data as needed. AJAX allows small HTTP requests to be sent in the background, thus never requiring a refresh of the webpage which makes the interface more responsive, and generally provides a better user experience. Using AJAX also helps keep memory demand on the web browser low because it does not necessitate the download of an entire data-source at once. The AJAX requests carry SPARQL queries to the aforementioned SPARQL server, which then returns relatively small chunks of data to the interface for rendering. Future improvements to the Pathway Browser will employ AJAX to incorporate precaching of likely future pathway selections.

Query results are returned in JSON to more closely mesh with the Javascript user interface. We chose JSON over SPARQL XML because returning results in XML required a Javascript based parsing algorithm to make use of them. JSON, on the other hand, offers the benefit of not requiring any parsing algorithms when being used within Javascript. Javascript can evaluate JSON as a native object and use it as a nested array-like structure, which is far simpler to work with.

As a result of these decisions, the GlycoBrowser client interface is only loosely tied with the data services it relies on, thus allowing them to be easily swapped out if necessary. Also, the browser retains platform independence by utilizing web technologies, and achieves greater efficiency through the use of a lightweight client application.

6 Conclusions and Future Work

Although there are many pathway visualization tools available, GlycoBrowser, a metabolic pathway exploration tool, is particularly suitable for glycobiologists. While knowledge, structures, and pathways are represented in the form of OWL ontologies, the graphical presentation is automatically constructed using molecule representations widely accepted by glycobiologists. Experimental data is overlaid in a contextually meaningful way. For example, as shown in Fig. 6, GlycoBrowser positions the glycomic and transcriptomic data in a way that facilitates the identification of correlations between these datasets as embryonic stem cells differentiate.

In the near future we plan to expand our toolset to include curation tools, allowing us to directly modify the GlycO and EnzyO ontologies to dynamically add new structure and pathway representations. We plan to further enhance the capabilities of

GlycoBrowser to simultaneously explore multiple pathway branches for comparison purposes. We also plan to include search capability for entire pathway fragments, in addition to currently available incremental exploration. Furthermore, we plan to add comprehensive filtering of overlaid experimental data.

References

1. Joseki, http://www.joseki.org
2. Open Biomedical Ontology
3. Aurrecoechea, C., Heiges, M., Wang, H., Wang, Z., Fischer, S., Rhodes, P., Miller, J., Kraemer, E., Stoeckert, C.J., Roos, D.S., Kissinger, J.C.: ApiDB: Integrated resources for the apicomplexan bioinformatics resource center. Nucleic Acids Research 35, D427–D430 (2007)
4. Brown, M.R.: FastCGI: A High-Performance Gateway Interface. In: Programming the Web - a search for APIs workshop. 5th Int. World Wide Web Conference, Paris, France (1996)
5. Carroll, J.J., Dickinson, I., Dollin, C., Reynolds, D., Seaborne, A., Wilkinson, K.: Jena: Implementing the Semantic Web Recommendations. In: Proc. of the 13th Int. World Wide Web Conference (WWW), pp. 74–83 (May 2004)
6. Ceroni, A., Dell, A., Haslam, S.M.: The GlycanBuilder: A fast, intuitive and flexible software tool for building and displaying glycan structures. Source Code for Biology and Medicine 2(3) (2007)
7. Dahlquist, K.D., Salomonis, N., Vranizan, K., Lawlor, S.C., Conklin, B.R.: GenMAPP, a new tool for viewing and analyzing microarray data on biological pathways. Nature Genetics 31, 19–20 (2002)
8. Ellson, J., Gansner, E., Koutsofios, L., North, S.: GordonWoodhull: Graphviz - Open Source Graph Drawing Tools. Lucent Technologies/AT&T Labs Research Tech. Report (2000)
9. Goldberg, D., Sutton-Smith, M., Paulson, J., Dell, A.: Automatic annotation of matrix-assisted laser desorption/ionization N-glycan spectra. PROTEOMICS 5, 865–875 (2005)
10. Harris, M.A., Clark, J.I., Ireland, A.: The Gene Ontology (GO) project in 2006. Nucleic Acids Research 34, D322–D326 (2006)
11. Janik, M., Kochut, K.: BRAHMS: A WorkBench RDF Store And High Performance Memory System for Semantic Association Discovery. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, Springer, Heidelberg (2005)
12. Karp, P.D., Riley, M., Paley, S., Pellegrini-Toole, A.: The MetaCyc Database. Nucleic Acids Research 30, 59–61 (2002)
13. Kelder, T., Pico, A.R., Iersel, M.v., Conklin, B., Evelo, C.: WikiPathways: Pathway Editing for the People. In: 11Th Annual International Conference on Research in Computational Biology, SanFrancisco (April 2007)
14. Klyne, G., Carroll, J.: Resource Description Framework (RDF). W3C recommendation (February 2004)
15. Kochut, K., Janik, M.: SPARQLeR: Extended SPARQL for Semantic Association Discovery. In: Franconi, E., Kifer, M., May, W. (eds.) ESWC 2007. LNCS, vol. 4519, pp. 145–159. Springer, Heidelberg (2007)
16. McGuinness, D.L., Frank van Harmelen, e.: Web Ontology Language (OWL). W3C recommendation (February 2004)

17. Nikitin, A., Egorov, S., Daraselia, N., Mazo, I.: Pathway studio—the analysis and navigation of molecular networks. Bioinformatics 19, 2155–2157 (2003)
18. Nimmagadda, S., Basu, A., Eavenson, M., Han, J., Janik, M., Narra, R., Nimmagadda, K., Sharma, A., Kochut, K.J., Miller, J.A., York, W.S.: GlycoVault: A Bioinformatics Infrastructure for Glycan Pathway Visualization, Analysis and Modeling. In: Proceedings of the 5th International Conference on Information Technology: New Generations (ITNG 2008), Las Vegas, Nevada (April 2008)
19. Ogata, H., Goto, S., Sato, K., Fujibuchi, W., Bono, H., Kanehisa, M.: KEGG: Kyoto Encyclopedia of Genes and Genomes. Nucleic Acids Research 27, 29–34 (1999)
20. Prud'hommeaux, E., Seaborne, A.: SPARQL: A Query Language for RDF. W3C recommendation (June 2007)
21. Rubio, D.: An Introduction to JSON (2007), http://dev2dev.bea.com/pub/a/2007/02/introduction-json.html
22. Sahoo, S.S., Thomas, C., Sheth, A., York, W.S., Tartir, S.: Knowledge Modeling and its Application in Life Sciences: A Tale of two Ontologies. In: World Wide Web Conference [WWW 2006], Edinburgh, Scotland (2006)
23. Saraiya, P., North, C., Duca, K.: Visualizing biological pathways: Requirements analysis, systems evaluation and research agenda. Information Visualization, 1–15 (2005)
24. Shannon, P., Markiel, A., Ozier, O., Baliga, N.S., Wang, J.T., Ramage, D., Amin, N., Schwikowski, B., Ideker, T.: Cytoscape: A software environment for integrated models of biomolecular interaction networks. Genome Res. 13, 2498–2504 (2003)
25. Thomas, C., Sheth, A.P., York, W.S.: Modular Ontology Design Using Canonical Building Blocks in the Biochemistry Domain. In: Proc. of the 4th Int. Conference on Formal Ontology in Information Systems (FOIS), Baltimore (November 2006)
26. York, W.S., Kochut, K.J., Miller, J.A., Sahoo, S., Thomas, C., Henson, C.: GLYDE- II – GLYcan structural Data Exchange using Connection Tables. University of Georgia Technical Report (2007)

Pattern Matching in RNA Structures

Kejie Li, Reazur Rahman, Aditi Gupta, Prasad Siddavatam, and Michael Gribskov*

Department of Biological Sciences, Purdue University, Lilly Hall of Life Sciences,
915 W. State Street, West Lafayette, IN 47907-2054, USA
gribskov@purdue.edu

Abstract. RNA plays key roles in many biological processes, and its function depends largely on its three-dimensional structure. We describe a comparative approach to learning biologically important RNA structures, including those that are not the predicted minimum free energy (MFE) structure. Our approach identifies the greatest conserved structure(s) in a set of RNA sequences, even in the presence of sequences that have no conserved features. We convert RNA structures to a graph representation (XIOS RNA graph) that includes pseudoknots, and mutually exclusive structures, thereby simultaneously representing ensembles of RNA structures. By modifying existing algorithms for maximal subgraph isomorphism, we can identify the similar portions of the graphs and integrate this with MFE structure prediction tools to identify biologically relevant near-MFE conserved structures.

1 Introduction

RNA molecules perform a variety of important biological functions in addition to carrying information from the chromosome to the ribosome, or acting as structural scaffolds. Catalytic RNAs play key roles in translation, RNA processing and splicing, and gene regulation [1]. Motifs that are important for RNA function are structural and correspond to base-paired regions of secondary structure, which in turn, provide the scaffold for the three-dimensional fold of the RNA [2, 3]. RNA sequences that have the same structural motifs may have sequences that are impossible to align because they have no detectable sequence similarity.

While programs that predict RNA secondary structure have been available since the 1980s, RNA structure prediction is handicapped by both biochemical and computational limitations. Firstly, RNA exists as an ensemble of rapidly interconverting structures. Protein structures (usually) show relatively minor fluctuations from a single minimum free-energy state. The case is much different for RNA where there are usually many structures with similar free-energies; these structures may be distinctly different in terms of base-pairing [4, 5]. Secondly, while we know that pseudoknot

* Corresponding author. Tel.: +1 7654946933; Fax: +1 765-496-1189.

I. Măndoiu, R. Sunderraman, and A. Zelikovsky (Eds.): ISBRA 2008, LNBI 4983, pp. 317–330, 2008.

structures are very important in RNA structure and catalytic function [6], it remains difficult to reliably predict pseudoknotted structures. This is due both to our incomplete understanding of the energetics of pseudoknot formation, as well as to the computational time complexity. The most efficient pseudoknot prediction algorithms, *e.g.*, pknotRG, have $O(n^4)$ time for certain classes of RNAs[7]), but achieve this by placing significant limitations on which structures can be found. Memory complexity of RNA structure prediction is $O(n^2)$, where n is the length of the RNA sequence, and usually ranges from 10,000-100,000 bases for primary RNA transcripts.

In biology, functionally important features can often be recognized because they are conserved over evolutionary time. A common approach is to obtain a set of sequences using some biological criterion (such as similarity of regulation), and use pattern recognition methods to identify unusually conserved features. Searching for sequence motifs (approximately common substrings) in this way has been a powerful tool for analysis of DNA and proteins; this approach does not work as effectively with RNA because conserved RNA structures may have no detectable sequence similarity. And while great progress has been made, it remains difficult to accurately predict MFE structures for RNA sequences. To further complicate the picture, RNAs exist as ensembles of structures, in addition to the MFE structure, that are constantly interconverting and fluctuating. The biologically important structures (those that are conserved over evolutionary time) may be present only transiently, or as minor components of this structural ensemble. The problem is further complicated by the fact that biology is messy; one can rarely get completely clean sets of sequence data in which every sequence actually contains the structure of interest. This makes many approaches unfeasible. In addition, in biological systems, conservation is only approximate, no set of structures will exactly match.

We are building a system that allows one to find the greatest approximately conserved structure(s) in a set of RNA sequences, in the presence of extraneous sequences that do not share a common structure. This conserved common structure can then be used as the basis for hypotheses about the importance of the structure in the biological functioning of the RNA. These hypotheses can be tested either experimentally or by further computational work.

We convert RNA structures to a graph representation that specifically includes pseudoknots and is capable of representing an ensemble of RNA structures in a single graph. Computationally, finding conserved structures corresponds to finding the greatest approximately isomorphous subgraphs in a set of graphs, where each graph represents a single RNA sequence. We use modifications of existing maximal subgraph isomorphism algorithms to identify the similar portions of the graphs, and propose to combine this with constrained MFE structure prediction tools [8], and a database search capability.

Graph theoretical approaches have previously been applied to RNA structures [9, 10], but our approach differs significantly. The XIOS approach introduces the ability to represent ensembles of structures, and emphasizes the topology of stems. Our approach is most similar to that of Gan *et al.*, but focuses on stem topologies rather than

the topology of loops and bulges [9]. The XIOS approach also allows structural motifs to be exactly matched without using heuristics [10].

2 XIOS RNA Graphs

In this section, we describe the graph framework that we have developed to represent ensembles of RNA structural topologies. We introduce the XIOS RNA graph representation for RNAs, and discuss extensions to existing subgraph isomorphism algorithms as they are apply to XIOS RNA graphs.

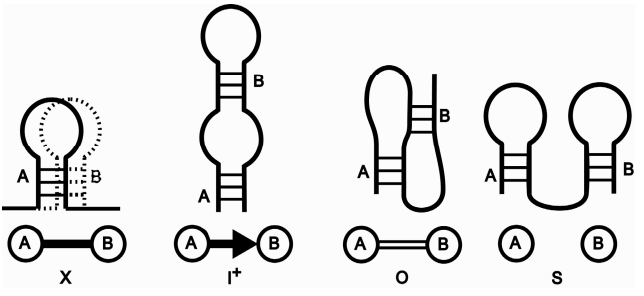


Fig. 1. XIOS definition. Relationships (edges) are defined as X (exclusive), I (included), O (overlapping), and S (serial). I^+ indicates the direction of I edges with respect to the higher numbered vertex and I^- indicates the opposite.

2.1 Definition

XIOS RNA graphs represent ensembles of RNA structural topologies. In XIOS graphs, each base-paired stem is represented by a vertex, and the edges connecting the vertices indicate the topological relationship between the stems. Topologically, two stems can be **eXclusive** (*i.e.*, both cannot simultaneously form because they use the same sequence ranges), **Included** (*i.e.*, one is nested within the loop of the other), **Overlapping** (*i.e.*, the stems have a pseudoknot relationship) or **Serial** (*i.e.*, adjacent, non-overlapping stem and loop structures) (Fig. 1). Each pair of vertices is related by one and only **X**, **I, O** or **S** relationship.

2.2 Training Data

We have developed Perl packages that translate Vienna RNA format [11] and the MFOLD [12] connect format into XIOS graphs. Because the predicted MFE structure is only one structure in a structural ensemble, we enumerate all energetically favorable short stems and label the entire set as X, I, O, and S, as described above. The graph is therefore an image of the entire structural ensemble. Our test datasets are described in Table 1. Highly similar sequences with sequence identity >40% are removed from the dataset to avoid selection bias.

Table 1. Brief description of RNA datasets. Formats are: A, alignment; C, MFOLD connect; S, sequence; V, Vienna RNA package.

Type of RNA	Database or Program	Format	Link
microRNA	miRNA	S	http://microrna.sanger.ac.uk/sequences/index.shtml
5S rRNA	Database	S	http://biobases.ibch.poznan.pl/5SData/
rRNA	RDP II	A, S	http://rdp.cme.msu.edu/index.jsp
RNase P	RNase P Database	C	http://www.mbio.ncsu.edu/RNaseP/
snoRNA	snoRNABase	S	http://www-snorna.biotoul.fr/
snoRNA	Plant snoRNA Database	A, S	http://bioinf.scri.sari.ac.uk/cgi-bin/plant_snorna/home
snoRNA	Human snoRNA Database	S	http://www.trex.uqam.ca/~snorna/Seqs.html
tRNA	GtRNAdb	V	http://lowelab.ucsc.edu/GtRNAdb/
tmRNA	tmRNA	A, S	http://www.indiana.edu/~tmrna/
Noncoding RNA	ncRNA Database	S	http://biobases.ibch.poznan.pl/ncRNA/
All	Pseudobase	V	http://biology.leidenuniv.nl/~batenburg/PKB.html
All	RNAbase	S	http://www.rnabase.org/
All	Rfam	A, S	http://www.sanger.ac.uk/Software/Rfam/index.shtml
All	RNAfold/MFOLD	C, V	Installed on local server

2.3 DFS Lexicographical Ordering

DFS (Depth-First Search) lexicographical ordering was originally developed by Yan et al. [13, 14] in their gSpan algorithm for identifying common chemical structures in chemical datasets. In the chemical structure case, both the vertices (atoms) and edges (bonds: single, double and triple) are labeled, and all edges are undirected. gSpan is a powerful search algorithm that reduces the search space for isomorphous subgraphs using a clever depth first search (DFS) preordered search tree.

The traversal order of edges and vertices in the DFS of a graph can be canonically ordered. This is called the *DFS tree*, or when serialized, the *DFS code.* Yan et al. proved that graphs with the same DFS code are, by definition isomorphous. Lexicographic rules provide an unambiguous best order to the canonical DFS code [13].

The direct path from the first traversed vertex (*root*) to the most recently added vertex (*right-most vertex*) is the *right-most path*. The extension of DFS graphs by edge growth is restricted to extension from the rightmost path, similarly to the approach of TreeminerV [15]. Graphs are extended in the following order: edges to existing vertices (backward edges), edges to new vertices extending from the right-most vertex, and extension from internal vertices on the right-most path. An intrinsic property of the DFS lexicographical ordering is that it creates a preorder that can be used to efficiently explore the search tree when searching for isomorphous subgraphs. Isomorphic forms of a graph fall in different positions in the search tree, but the canonical DFS representation of a particular isomorph is guaranteed to be found first. Hence, the lexicographically first instance of an isomorph in the search tree is its *minimum representation* or *canonical labeling* and other instances can be efficiently pruned. Each edge in the DFS code is described by a 3-tuple, $(v_i, v_j, l_{i,j})$, where v_i and v_j are two connected vertices and $l_{i,j}$ is the label of the edge. Fig. 3 shows how the canonical

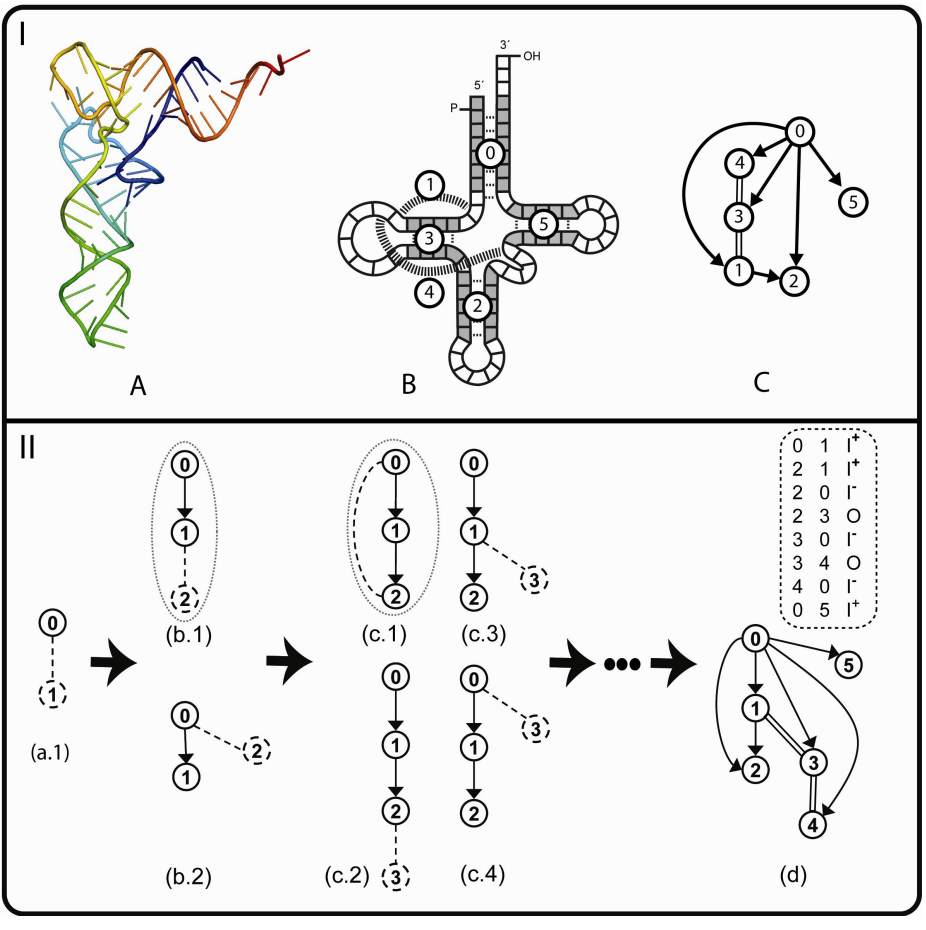


Fig. 2. tRNA 3D structure and corresponding XIOS graph representation. I.A. 3-D structure of tRNA (PDB ID, 1EHZ). I.B, the simple three-leaf clover shape of tRNA is shown, where the acceptor stem, D-arm, anticodon-arm, and T-arm are represented by vertices 0, 3, 2 and 5 respectively. Vertex 1 represents an interaction between the D-loop and a region between the D-arm and acceptor-arm, and vertex 4 represents an interaction between the D-loop region and the region between anticodon-arm and T-arm. In the XIOS representation (I.C), vertex 1 is included in the acceptor stem and overlaps with the D-arm, vertex 4 overlaps with the D-arm and the Anticodon arm is included in vertex 4. II a, b, and c show the sequential extension of the DFS graph, and II d shows the minimum DFS tree and corresponding DFS code. At the each stage of graph extension, all the possible extensions are shown in dotted lines. For each edge extension, only the canonical graph (shown by dotted ellipse) is used in the next stage.

labeling can easily be identified using lexicographic rules even though many different DFS codes are possible. There are two additional rules that prune the search space. Firstly, if the initial edge of a minimum DFS code is type e_0, then no following edge can have a lexically smaller edge label, and secondly, for any backward edge growth

to v_j, an edge cannot be lexically smaller than any edge that is already connected to v_j or $v_{rightmost}$ [13]. Each distinct mapping of vertices to a DFS code is the *support* for that potential solution. Since many such mappings are possible, each graph may have multiple support for a DFS code. As a simple example, Fig 2 shows the XIOS graph for a tRNA, according to the experimentally determined 3-dimensional structure (PDB ID: 1EHZ).

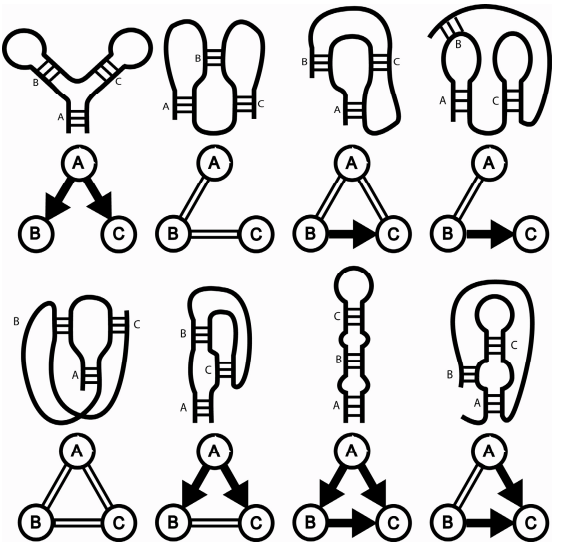


Fig. 3. Unique three-stem XIOS graphs, including pseudoknots. Fifteen XIOS graphs with three vertices are possible, three of them are not true three-stem topologies (at least one of the stems has only S relationships with other stems); the other four three-stem structures are either redundant or physically impossible.

2.4 Enumeration N-stem structures

Every RNA structure can be represented by a XIOS graph. For *n* stems, the upper bound on the number of possible structures[1], *N*, can be calculated by Equation (1),

$$N = \frac{(2n)!}{2^n \cdot n!} \tag{1}$$

For example, there is only one possible one-stem structure, two possible two-stem structures, and 10 possible three-stem structures, but only eight unique structures

[1] For the *n*-stem case, there are *2n* half stems. We assign integer labels to each half stem from *1* to *2n-1*.By definition, the first half stem is labeled 1, and there are *2n-1* possible half stems that can pair with the first half stem; the third half stem has only one possible label (2 or 3), and there are *2n-3* possible half stems that can pair with this half stem, and so on. The upper boundary of the number of possible n-stem structures is therefore: *(2n-1)*(2n-3)*(2n-5)*…*5*3*1.*

(Table 2). Fig. 3 shows the XIOS graphs for the eight unique structures that can be formed from three stems. The other two three-stem structures are either redundant or physically impossible.

Table 2. Number of possible RNA topologies for different numbers of stems, N

N	Total	Unique Topologies	% Unique
1	1	1	100
2	2	2	100
3	10	8	80.00
4	78	49	61.25
5	746	389	52.14
6	8566	4207	49.11
7	114834	56227	48.96

3 Greatest Conserved Structures

3.1 Extension of the gSpan Algorithm

XIOS graphs have several differences from the chemical structure graphs considered by Yan and Han. XIOS graphs

- have both directed and undirected edges. *I* edges are directed because it is highly important whether a stem is nested within or outside another stem. *X*, *O*, and *S* edges are undirected.
- do not have vertex labels. Because every vertex is simply an anonymous elemental stem, no labels are available.

The use of unlabeled vertices with the gSpan algorithm is fairly straightforward, but results in a decreased ability to rapidly prune the search tree. Directed edges are a little more difficult to accommodate because the direction of the edge depends on the vertex from which one looks. The simplest approach is to label the edge as either I^+ or I^- from the point of view of the lowest numbered vertex. I^+ and I^- are treated as lexicographically distinguishable edges.

In the original application of gSpan to chemical structures, Yan and Han were interested in identifying frequently occurring chemical substructures. In their case, structures that occur many times in a single graph are equally interesting. The case of RNA differs; motifs that occur in multiple graphs (molecules), rather than many times in a single graph (molecule), are considered more important. In addition, the presence of incorrectly classified sequences, *i.e.*, sequences that have no common structure, means that not all graphs will support the biologically relevant subgraph. For XIOS graphs, therefore, support is calculated as the number of graphs that containing a subgraph, rather than the total count of matching subgraphs.

3.2 Graph Matching Algorithm (Similar to *gSpan*[2])

```
begin:
For a XIOS graph G with edges e_G
I.  Sort edges in e_G by edge type e_G ∈ {X,I,O,S}
II. For each edge type E
    1.  Find all lexicographically minimal one edge
        subgraphs, S, from the given XIOS graphs;
    2.  For each edge e in S
    3.  Do Subgraph_mining(G, S, e):
          i.   If the graph is NOT a minimum graph ac-
          cording to DFS lexicographical order, return;
          ii.  Generate all potential children with one
          edge growth, e_new
          iii. If support for each child is above
          threshold
    Recursively call Subgraph_mining with updated edge
    list (G, S+, e_new)
    4.  Remove all edges of edge type E from G after
    all descendents have been searched
    5.  If e_G = Ø, break;
end.
```

3.3 Greatest Conserved Structure(s) in a Set of RNAs

Many computational approaches use pairwise or multiple DNA or protein sequence alignments to find conserved motifs, but this approach is generally impossible with RNA sequences because of their lack of conserved sequences, and because of the difficulty of obtaining unambiguously correct alignments. However, secondary and higher order structures in RNA are conserved, so matching the topology of two RNA structures with a graph matching approach can identify conserved motifs that cannot be seen in the sequences. The pre-ordered DFS search approach of gSpan provides an effective approach to this problem.

The time complexity for the worst case of this algorithm is suggested to be *O(kn)* [13, 14], where *k* is the maximum number of subgraph isomorphisms existing between the two graphs and *n* is the size of the greatest common match. Fig. 4 shows the application of the XIOS graph approach to the structure of *S. cerevisiae* and *H. sapiens* RNase P.

3.4 Characteristics of Biological Graphs

The graph isomorphism approach is limited by the size of the graphs. We examined sequences from snoRNA, 5S rRNA, microRNA, tRNA, and RNase P (See Appendix for details) to determine how the number of stems varies with sequence length in biological RNAs. The sequences were obtained from online databases (Table 1) and their predicted MFE structures were obtained using the RNAsubopt program of the Vienna

[2] Adapted from [13] with minor modification.

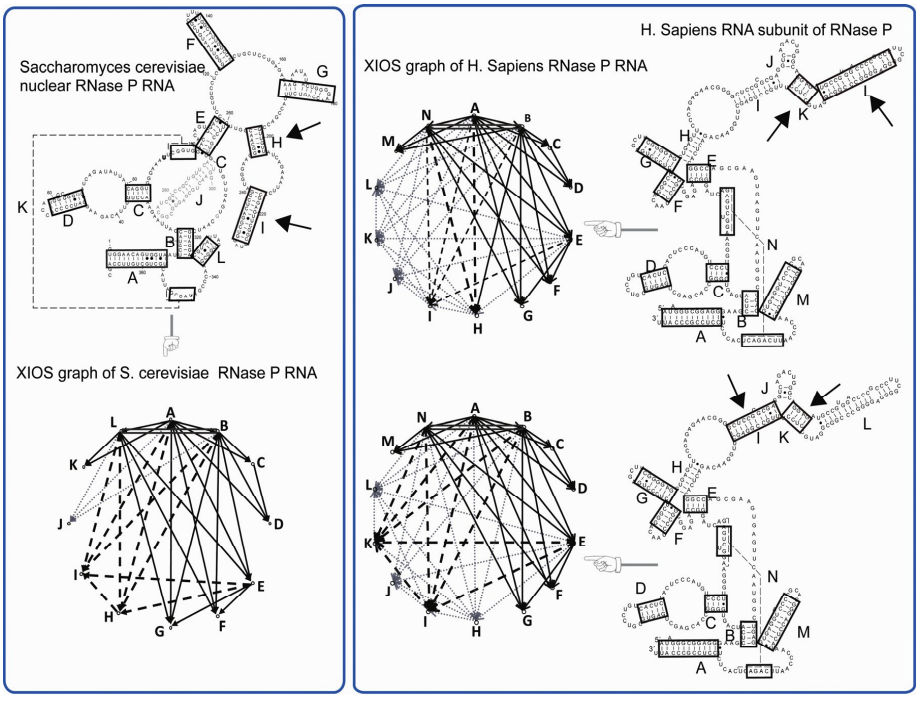


Fig. 4. Identification of the common structure in *S. cerevisiae* and *H. sapiens* RNase P RNA. Left panel (top) shows the secondary structure of the *S. cerevisiae* RNAse P RNA. Each stem is labeled with a capital letter A-L. Left panel, bottom, shows the XIOS graph. I edges are shown as single lines and O edges as double lines. Right panel shows the secondary structure (A-N) and XIOS graphs for a single human RNAse P RNA. In both panels, matching secondary structures are enclosed by boxes and the uniquely matching part of the XIOS graphs shown in dark lines. Dotted lines in the XIOS graphs indicate where there are multiple mapping between stems H and I of the *S. cerevisiae* structure and the human structure; these multiple mapped stems are also indicated by arrows in the secondary structure diagrams. The right panel shows two of the mappings as an example.

RNA package [5]. Predicted MFE structures were also obtained for random sequences in a similar fashion. Random sequences were obtained by randomizing the order of bases in the corresponding biological sequences, thus preserving the base composition and sequence length.

Fig. 5 indicates the overall trend of linear increase in number of stems as a function of sequence length. This rapid increase in the number of stems is due to the intricately folded structures of the RNAs. This observation further necessitates the development of an efficient system for searching biologically relevant structural patterns in RNA. It is notable the biological RNAs and random RNAs have very similar numbers of structures. As one can see in fig. 6, stem structures in biological RNAs are predominantly less than ten base-pairs long.

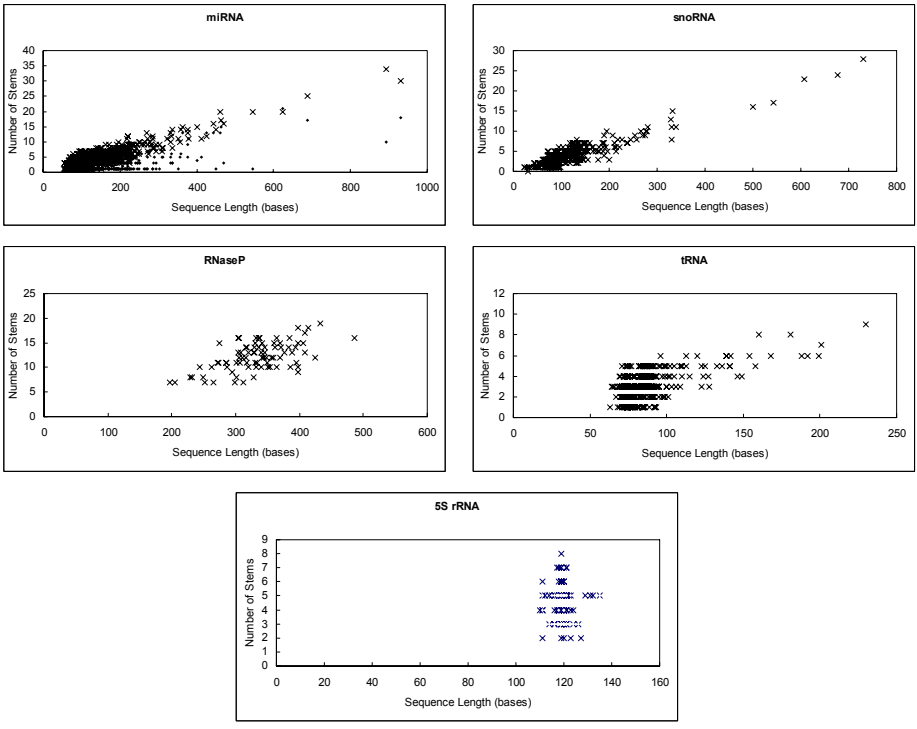


Fig. 5. Correlation between number of stems and sequence length. Number of stems in biological (♦) and randomized (×) RNA sequences versus sequence length. The number of stems increases roughly linearly with sequence length. Each biological sequence was permuted to generate a corresponding random sequence, preserving the sequence length and base composition of the real sequence.

4 Future Directions

The number of stem structures in an RNA MFE structure can be very large (Fig 5); the total number of possible stems, however, grows quadratically with the length of the sequence. If one assumes that stem-loop structures require on average 24 bases, the number of possible stems would be something like $(SequenceLength/24)^2$. For a relative short 10kb mRNA sequence this would lead to graphs with over 150,000 vertices. Our ultimate goal is to analyze 10-20 sequences of much longer length (many biological RNAs are over 100,000 bases long), a daunting problem. There are a number of approaches that can be used to reduce the size of the problem. These include preprocessing the structure to include only the most interesting stems (rather than all possible stems), the application of graph contraction methods, and the introduction of vertex labels.

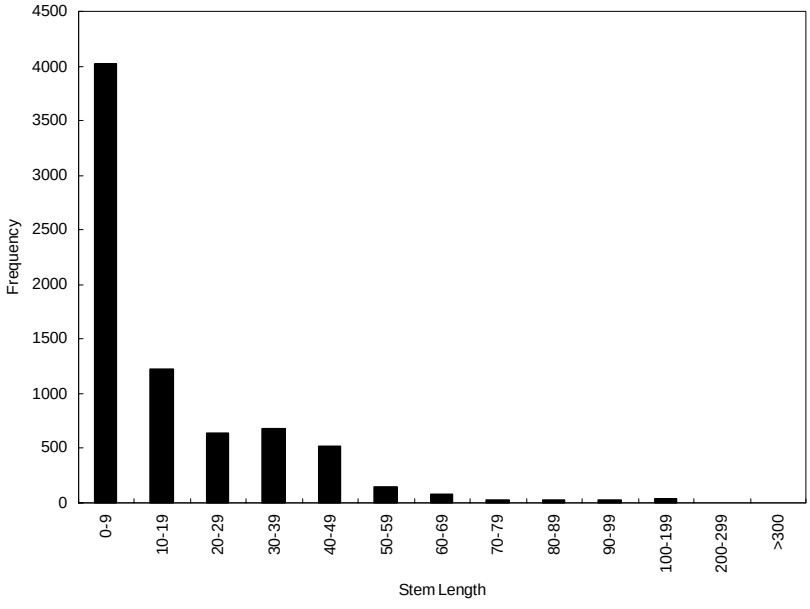


Fig. 6. Length of RNA stem structures in biological RNAs

4.1 Graph Preprocessing

While the most biologically interesting RNA structure need not be the minimum free energy (MFE) structure, it is likely that the important structures are close to the MFE [16]. This follows from the Boltzmann relationship, which indicates that the relative frequency of a given structure in the structural ensemble depends on its energy. Rather than identifying all short energetically favorable stems, we can greatly reduce the size of the problem by including only stems that participate in a structure within some energy interval, ∂, from the predicted MFE structure. The total number of stems can be controlled by altering ∂; ∂=0 produces the MFE structure.

4.2 Reduction of Graph Complexity

Graph contraction reduces graph complexity by pruning irrelevant vertices and edges. There are a number of different approaches one can take to pruning XIOS graphs. Firstly, as we pointed out above, one can simply discard the S edges; since there are exactly four edge types and each pair of vertices has exactly one edge, only three edge types need be used. Secondly, we can place limits on the construction of edges of other types, especially of *I* edges. One of the advantages of the XIOS representation is that nested stems, represented by *I* edges, have an edge with every other stem in which they are included. This embedding can be many levels deep, generating a huge number of highly connected vertices. This is a great advantage because it obviates the need for introducing gaps [17] which make the matching problem much more complex (and *ad hoc* since there is no way to determine correct gap parameters). We

postulate that we would lose little matching power if the depth of *I* edge nesting was limited to a fixed depth such as four. This would still permit extraneous stems to be easily omitted but greatly reduce the number of edges in the graphs. Finally, because we can enumerate all possible XIOS structures with a fixed number of stems, we can create a dictionary of these substructures and condense the graphs to a smaller number of vertices based on this dictionary, at the same time converting the unlabelled vertices to labeled vertices (the labels then correspond to the dictionary structures).

4.3 Adding Labels

The dictionary strategy, described above, faces difficulties since the isomorphous structure of interest is buried in a huge field of random noise. If the dictionary based labels are dominated by the non-matching (noise) portion of the graph, the re-encoded graph will lose the information needed to match to other graphs (*e.g.*, if the dictionary structures overlap but do not exactly correspond to the interesting conserved structures). A similar strategy, unique to the XIOS graph, is to examine all three vertex triangles, of which there are a strictly limited number of types due to the limitations both of the graph and of the biochemistry of RNA, and replace each triangle with a corresponding labeled vertex. Triangles may share one or two edges which can be incorporated as an extended set of edge labels. Such graphs would be modestly smaller, but much more heavily labeled, greatly increasing the search speed. At the same time, little information is lost since the original graph can be almost completely reconstructed from the triangle-condensed graph.

4.4 Motif Identification Tool

RNAs that interact with specific molecules, such as proteins, generally have common topological motifs. For example, in alternative splicing the donor, acceptor, and branch point all have specific conserved structures important in recognition and catalysis. Such conserved structures, when identified in molecules of unknown function, immediately generate experimentally testable hypotheses. Once motifs are identified, they can be used to search for additional sequences that could form the same structure. This provides a means for both statistically evaluating the significance of the structural motif, as well as for validating matches by examining them for biological similarities, *e.g.*, by comparing the GO annotations [18] of the sequences. A number of approaches may be suitable for this, including stochastic context free grammars (SCFG) [19] which are frequently used to identify RNA structures based on biological knowledge [20].

4.5 Database Search Tool

For searching of large databases, SCFGs are likely to be too slow. We are developing a fast database search tool for RNA motifs. Since we can enumerate all possible XIOS graphs up for structures of up to 7 or 8 stems (hundreds of thousands) we believe that we can use the enumerated structures to prescreen graphs in much the same way that BLAST [21] uses identically matching words. This is closely related to the dictionary concept introduced above. Because matching to the enumerated structures in the dictionary can be precalculated, we plan to develop a fast system based on the

observation that one need not do the complete isomorphous subgraph search if two sequences share no dictionary motifs, and that if they do, the isomorphism search can be seeded by the matching motifs. Such a search tool would allow users to both extend and validate motifs found through subgraph isomorphism matching, and would also provide a means to functionally classify unknown RNAs. RNA is still rather poorly understood and such an approach will be of great use in identifying novel structural and functional motifs.

Because RNA structures are relatively degenerate, it is likely that a post-processing system will be needed to identify the most interesting possible structures out of a large number of possibilities. This issue is similar to the problem of relevance ranking in web indexing. In sequence comparisons, statistical probability calculations are commonly used as a relevance ranking mechanism, and this may be possible in the XIOS system; we anticipate that the distribution of maximal matching structures will follow an extreme value distribution. Any two large RNAs, however, will have common structures that are almost completely trivial: they will match as a long series of serial stems. This is generally not biologically interesting, suggesting that there is a notion of biological complexity which can be used as a relevance ranking function. This biological notion of complexity may or may not correspond to mathematical notions of graph complexity [22]. Another possible relevance function would be to choose only structural motifs that can form near-MFE predicted structures using a constrained folding approach (motif stems are constrained to base-pair in the predicted structure) such as are available in MFOLD and the Vienna RNA package.

The XIOS graph representation has great promise for identifying biologically interesting structural motifs in RNA based on sequence alone. Constructing a sufficiently fast motif search system will allow RNA studies to take advantage of the same bootstrap process that is commonly used for DNA and protein sequences, namely 1) identify biologically related sequences, 2) identify statistically significant structural motifs, 3) use structural motifs to identify additional candidate sequences (iterating to convergence), and 4) use the structural motif as a basis for laboratory experiments.

Acknowledgments

This work was supported by National Science Foundation award DBI-0515986.

References

1. Birney, E., Stamatoyannopoulos, J.A., Dutta, A., Guigo, R., Gingeras, T.R., Margulies, E.H., Weng, Z., Snyder, M., Dermitzakis, E.T., Thurman, R.E., et al.: Identification and analysis of functional elements in 1% of the human genome by the ENCODE pilot project. Nature 447(7146), 799–816 (2007)
2. Zarrinkar, P.P., Williamson, J.R.: The kinetic folding pathway of the Tetrahymena ribozyme reveals possible similarities between RNA and protein folding. Nature structural biology 3(5), 432–438 (1996)
3. Doherty, E.A., Doudna, J.A.: The P4-P6 domain directs higher order folding of the Tetrahymena ribozyme core. Biochemistry 36(11), 3159–3169 (1997)

4. Zuker, M.: On finding all suboptimal foldings of an RNA molecule. Science 244(4900), 48–52 (1989)
5. Wuchty, S., Fontana, W., Hofacker, I.L., Schuster, P.: Complete suboptimal folding of RNA and the stability of secondary structures. Biopolymers 49(2), 145–165 (1999)
6. Staple, D.W., Butcher, S.E.: Pseudoknots: RNA structures with diverse functions. PLoS biology 3(6), 213 (2005)
7. Reeder, J., Giegerich, R.: Design, implementation and evaluation of a practical pseudoknot folding algorithm based on thermodynamics. BMC bioinformatics 5, 104 (2004)
8. Mathews, D.H., Disney, M.D., Childs, J.L., Schroeder, S.J., Zuker, M., Turner, D.H.: Incorporating chemical modification constraints into a dynamic programming algorithm for prediction of RNA secondary structure. Proceedings of the National Academy of Sciences 101(19), 7287–7292 (2004)
9. Gan, H.H., Pasquali, S., Schlick, T.: Exploring the repertoire of RNA secondary motifs using graph theory; implications for RNA design. Nucl. Acids Res. 31(11), 2926–2943 (2003)
10. Kim, N., Shiffeldrim, N., Gan, H.H., Schlick, T.: Candidates for Novel RNA Topologies. Journal of molecular biology 341(5), 1129–1144 (2004)
11. Ivo, L.F.H., Peter, F.S., Sebastian, B.L., Manfred, T., Peter, S.: Sebastian, Tacker Manfred, and Schuster Peter: Fast Folding and Comparison of RNA Secondary Structures. MonatshChem 125, 167–188 (1994)
12. Zuker, M., Stiegler, P.: Optimal computer folding of large RNA sequences using thermodynamics and auxiliary information. Nucleic acids research 9(1), 133–148 (1981)
13. Yan, X., Han, J.: gSpan: Graph-Based Substructure Pattern Mining. In: Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM 2002), p. 721. IEEE Computer Society, Los Alamitos (2002)
14. Yan, X., Han, J.: CloseGraph: Mining closed frequent graph patterns. In: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, Washington, D.C. ACM, New York (2003)
15. Zaki, M.J.: Efficiently mining frequent trees in a forest. In: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining, Edmonton, Alberta, Canada, ACM Press, New York (2002)
16. Jaeger, J.A., Turner, D.H., Zuker, M.: Improved predictions of secondary structures for RNA. Proceedings of the National Academy of Sciences of the United States of America 86(20), 7706–7710 (1989)
17. Wang, Z., Zhang, K.: Alignment between Two RNA Structures. In: Mathematical Foundations of Computer Science 2001, p. 690 (2001)
18. Ashburner, M., Ball, C.A., Blake, J.A., Botstein, D., Butler, H., Cherry, J.M., Davis, A.P., Dolinski, K., Dwight, S.S., Eppig, J.T., et al.: Gene ontology: Tool for the unification of biology. The Gene Ontology Consortium. Nature genetics 25(1), 25–29 (2000)
19. Grate, L., Herbster, M., Hughey, R., Haussler, D., Mian, I.S., Noller, H.: RNA modeling using Gibbs sampling and stochastic context free grammars. In: Proceedings / International Conference on Intelligent Systems for Molecular Biology; ISMB, vol. 2, pp. 138–146 (1994)
20. Lowe, T.M., Eddy, S.R.: tRNAscan-SE: A program for improved detection of transfer RNA genes in genomic sequence. Nucl. Acids Res. 25(5), 955–964 (1997)
21. Altschul, S.F., Gish, W., Miller, W., Myers, E.W., Lipman, D.J.: Basic local alignment search tool. Journal of molecular biology 215(3), 403–410 (1990)
22. Pudlák, P., Rödl, V., Savický, P.: Graph complexity. Acta Informatica 25(5), 515–535 (1988)

The Use of a Conformational Alphabet for Fast Alignment of Protein Structures

Wei-Mou Zheng

Institute of Theoretical Physics, Academia Sinica, Beijing 100080, China

Abstract. A protein conformational alphabet refers to the discretized states of the three-dimensional segmental structure of protein backbones. Here a letter corresponds to a cluster of combinations of three angles formed by C_α pseudobonds of four contiguous residues, and our alphabet consist of 17 letters obtained by clustering based on the probability distribution of these angles. A substitution matrix called CLESUM has been derived from an alignment database of representative structures to measure both evolutionary and geometrical similarity between any two such letters. A structural fragment is then mapped to a string, and two strings with their CLESUM score being higher than a preset threshold form a similar fragment pair (SFP). The search for SFPs by string comparison is fast. Furthermore, CLESUM scores reflect the importance of SFPs to structure alignment, and then the search space can be significantly reduced. A fast tool for pairwise alignment called CLePAPS is developed by collecting as many spatially consistent SFPs as possible. Extending the concept of SFPs to that of similar fragment blocks for multiple structure alignment leads to a fast tool for multiple structure alignment called BLOMAPS. Both CLePAPS and BLOMAPS are tested on ensembles of various structures. They are reliable, and about two or three orders faster than some well-known algorithms.[1]

1 Introduction

The comparison of protein structures has been an extremely important problem in structural and evolutional biology. Protein structure comparison is most often performed by a protein structure alignment program. Local similarity is a necessary condition for the global structural alignment, but insufficient. Structurally similar fragments first found in different proteins by seed matches form the basis objects for further examination of their consistency in the spacial arrangement required by the global alignment. Consistent pieces then may be joined to obtain the global alignment.

Biologically important modules have been repeatedly employed in protein evolution by gene duplication and rearrangement mechanisms. They form components of fundamental units of structure and function. The existence of such

[1] This work was supported in part by the National Basic Research Program of China (Grant No. 2007CB814800) and the National Natural Science Foundation of China.

I. Măndoiu, R. Sunderraman, and A. Zelikovsky (Eds.): ISBRA 2008, LNBI 4983, pp. 331–342, 2008.

conservative recurrent segments presents a solid foundation for the local analysis. Besides coordinates, distances and angles, another way to represent structures is to use conformational alphabets, which are discretized conformational states of certain fragment units of protein backbones. The smallest unit possessing one-to-one correspondence between angles and coordinates is the quadrupeptide unit, which admits two bending angles and one torsion angle. Our conformational alphabet of 17 letters is obtained by clustering based on the probability distribution in the phase space spanned by these three angles (Zheng and Liu, 2005). The description by conformational letters provides a good balance between accuracy and simplicity, and converts a 3D structure to a 1D sequence.

Without substitution matrices the use of conformational alphabets is very limited. In order to implement fast structural comparison by means of our conformational alphabet, we have derived one of such matrices called CLESUM from a representative pairwise aligned structure set of the FSSP (families of structurally similar proteins) database of Holm and Sander (1997). All the pair alignments of the FSSP for the proteins with a sufficient similarity in the representative set are collected for counting aligned pairs of conformational letters. An entry of the CLESUM is the log-ratio (with the base 2) of the observed frequency of an aligned letter pair to the expected frequency from a random alignment simply by chance. A scaling factor of 20 instead of 2 has been used to show more details. Taking the database statistics into account, CLESUM has a relatively low score for a match of frequent helix or sheet sites. It has been verified that CLESUM aptly measures the similarity between the conformational letter states (Zheng and Liu, 2005). Despite the existence of various methods for structure alignment, efficient and reliable algorithms for fast alignment are in ever increasing demand for analyzing the rapidly growing data of protein structures. Here we report tools developed for fast alignment of protein structures by fully using our conformational alphabet and its substitution matrix CLESUM.

2 CLePAPS: A Fast Pairwise Structure Alignment Tool

The common goal of all pairwise structure alignment methods is to identify a set of residue duads from each protein that are structurally similar, or to find the optimal correspondence between the atoms in the two molecular structures. An exhaustive search for such atomic correspondence is intractable, and various heuristics have been developed. For example, to lower the dimensionality of the problem, DALI identifies interaction patterns of fragment pairs (Holm and Sander, 1997), VAST describes secondary structure elements (SSEs) as a set of vectors (Gibrat et al., 1996) while CE designates short similar fragment pairs (SFPs) of local structural similarities (Shindyalov and Bourne, 1998).

For a given correspondence of two point sets, finding the best rigid transposition to superpose the correspondence sets can be easily done by using a closed-form solution based on the singular value decomposition (Kabsch, 1978). When the transformation between the two sets is given, the problem to find the correspondences (of ϵ-congruence at the maximal or average error tolerance ϵ)

is rather straightforward. However, when aligning two protein structures, at the beginning we know neither the transformation nor the correspondence.

Many methods start with an initial correspondence (seed matches), from which the optimal transformation for the correspondence is determined. The transformation is then used to update the correspondence. The procedure of progressively building up larger correspondence is iterated until the best correspondence is finally found. A typical example is ProSup (Lackner et al., 2000). Using the 3D segmental structure coding of our conformational alphabet, we develop a fast tool called CLePAPS for conformational letters based pairwise alignment of protein structures.

2.1 Similar Fragment Pairs of High Scores

Among various protein properties measuring structural similarity such as inter-residue distances or chemicophysical environments, CLePAPS uses a coarse-grained similarity between conformational letters. The kernel of a CLePAPS alignment consists of as many consistent similar fragment pairs (SFPs) as possible. CLePAPS regards an SFP as an ungapped string pair with a high sum of pairwise CLESUM scores. CLePAPS searches for SFPs by simple string comparison. Suppose that the pair of structures to be aligned is P and P' with P being the shorter. The coordinates $\{\mathbf{r}_i\}$ and $\{\mathbf{r}'_j\}$ of C_α atoms of the two proteins are converted to the sequences S and S' of conformational letters, respectively. Since each letter corresponds to a quadrupeptide unit, the length of S (S') is shorter than that of P (P') by 3. By convention, we assign the first letter to the third residue, the second to the fourth and so on, until finally the last letter is assigned to the last residue but one.

Consider two fragments of the same length l, one of which starts at residue i of P and the other at j of P'. The local structural similarity of the fragment pair may be measured by

$$\sigma = \sum_{k=0}^{l-1} M(s_{i+k}, s'_{j+k}), \tag{1}$$

where $M(a, b)$ is the (a, b)-entry of the CLESUM, and s_i and s'_j are the conformational letters of corresponding residues. Setting a threshold T, if the pair score $\sigma \geq T$, we call the pair an SFP, which defines a correspondence (l residue duads). The two members of an SFP are referred as neighbors of each other. Searching for SFPs by string comparison is fast. Furthermore, compared with the usual definition of SFP by purely geometrical similarity, our definition also gains specificity.

When an SFP contains residues of SSEs at its ends, shifts of the SFP often also form SFPs. To remove such redundancy, we keep only the one with the highest score among the nearby SFPs which are shifts of each other. A width w is set to restrict the maximum overlap for this 'shaving'. After shaving we have a reduced list of the representative SFPs. We sort the list in descending order of scores. Usually, a small l and a low T will result in a long list of SFPs.

For a given long enough SFP, we can find a rigid transformation to superpose its two members and make the spatial deviation of its duad C_α atoms very small. Since an SFP is determined only by local similarity, a superposition valid for one SFP need not be valid for another. We define the spatial deviation or separation between two members of a certain SFP after a transformation by

$$\delta = \max_{(\mathbf{r}_i, \mathbf{r}'_j) \in \text{SFP}} \{|x_i - x'_j|, |y_i - y'_j|, |z_i - z'_j|\}, \tag{2}$$

where $(\mathbf{r}_i, \mathbf{r}'_j)$ is a duad of the SFP after transformation, and (x, y, z) denotes the 3D coordinates of $\mathbf{r}$. A small separation δ implies a good superposition of the two SFP members.

2.2 The Greedy 'Zoom-in' Strategy

There is no clearly defined unique way to evaluate the quality of protein structure alignments. We adopt the standard of ProSup: the goal is to maximize the number N_e of structurally equivalent residues subject to a fixed Euclidean distance cutoff d_0 for judging correspondence between a residue duad and a minimal aligned segment size ρ. CLePAPS uses $d_0 = 5\,\mathring{A}$ and $\rho = 4$.

To balance speed with accuracy, we generate two lists of SFPs, one for $l = 20$ with threshold $T_{20} = 350$ and overlaping width $w = 20$, and the other for $l = 8$ with $T_8 = 0$ and $w = 4$. Any two helices are locally similar. Length 20 will exclude many such purely local coincidence. Length 8 is necessary for including most significant aligned pieces. We denote them as List-20 and List-8, respectively. The two lists can be generated in a single run. We expect that a significant alignment should contain at least one SFP of length around 20. Initial primary correspondences will be taken from the top ten SFPs of the sorted List-20. If the list size is less than 10, 'top ten' means all.

Once an SFP is chosen as an anchor, the transformation optimal to the SFP may be used to superimpose the two proteins. The separation δ of any SFP can then be calculated. Some SFPs are consistent with the anchor. That is, they will have a small separation δ. By thinking in terms of graph theory, the anchor or center and its consistent SFPs form a star tree or star. We define the size of a star as the total number of its SFPs.

Taking each of the top ten SFPs as an anchor, we find its consistent SFPs or neighbors in the top 50 SFPs of the sorted List-20. The stars really used by CLePAPS are subject to a further restriction: for a given anchor, we search the sorted List-20 successively from the top for neighbors of the anchor, and add a new neighbor SFP only when it does not overlap with any existing neighbor SFPs. In this way, we obtain ten restricted stars. We sort them first by their sizes, and then by similarity score σ of their centers in descending order. We remove the stars whose centers are neighbors of the first star. Then, we examine the next star, and remove the stars associated with its neighbors, and so on, until all stars are examined. Only the centers of the retained stars will be taken as an initial alignment seed. The effect of this star removal is twofold: removing seed redundancy and selecting the seeds which better reflect the global consistency.

The extension of an initial seed alignment is mainly done by the following blank-filling of the SFPs from List-8 which are consistent with the given anchor. Blanks are residue positions not included in an existing correspondence. The transformation optimal to the anchor SFP need not be globally optimal. We use a multi-step 'zoom-in' strategy, starting with a low precision to avoid local trapping. We first use a large cutoff $d_1 = 8\,\mathring{A}$ as the consistency criterion. That is, we add only the SFPs with $\delta < d_1$ to the existing correspondence. The procedure of blank-filling is greedy. The SFPs with a higher σ have a priority to be filled. We examine the top half of the SFPs in the sorted List-8. When blank-filling is fulfilled, the transformation optimal to the enlarged correspondence is determined to update the superposition of the two proteins. In the next run of blank-filling, cutoff d_1 is reduced to $d_2 = 6\,\mathring{A}$, and five-sixths of the SFPs in List-8 are examined. In a third run, d_2 is further reduced to $d_3 = 5\,\mathring{A}$, and the whole list is examined. Usually, three runs of iteration are enough for obtaining a full alignment.

There are mainly two ways to update the correspondence. One is to keep the existing duads and add new ones. The other is to re-start with an empty correspondence and then fill in blanks with SFPs from List-8. The latter strategy is used in CLePAPS. In the final polishing stage, the SFPs which have only a limited overlap with the existing correspondence can also be used for blank-filling. We speed up computation by means of marking. At the beginning, all SFPs in List-8 are identified as 'unmarked'. If an SFP has no two contiguous residue duads whose coordinate differences are both smaller than d_i, it will be marked as 'invalid', and then would never be examined again.

2.3 Refinement by Elongation and Shrinking

After blank-filling we obtain an alignment usually as disjunct pieces. Due to the finite size of SFPs and the redundancy removal by shaving, it is possible that an aligned piece can be elongated near its ends. If the Euclidean distance of an outer residue duad is smaller than d_0 we elongate the aligned piece by joining the duad to it. More nearby residue pairs can be further examined for elongation. On the other hand, depending on the relative quantities of d_0 and d_3, the Euclidean distance between some residue duads on the aligned pieces would be greater than d_0. We remove such duads from the alignment (as a shrinkage of the SFPs).

A filter for a minimal aligned segment length ρ is finally applied. A further iteration of transformation would additionally improve the quality of the alignment. Once a global alignment is accomplished, the total number N_e of equivalent residue duads and the RMSD of the alignment are calculated as quality indicators. Despite the star removal two alignments generated from two star centers may still be very similar. We compare entries of the rotation matrices. If the greatest difference between two corresponding matrix elements is below ε, say 0.1, the two alignments are regarded as identical. A more careful criterion is the rotation angle defined for two matrices (Vriend and Sander, 1991).

Structure comparison often yields several distinct alignments as multiple solutions. The existence of alternative alignments is mainly due to structure symmetry and repeats at different levels ranging from secondary structure, supersecondary structure to domains. Another source is the domain move. CLePAPS often reports several alignments and ranks them according to their N_e.

2.4 The Fischer Benchmark Test

A well-known comprehensive test set for assessing the performance of fold recognition methods is the benchmark of Fischer et al. (1996), which contains 68 pairs of proteins. All pairs of the set are known to be structurally similar, but they have low sequence identity. This set covers a wide range of protein families. We test CLePAPS on the benchmark. Ten protein pairs from the Fischer benchmark set were regarded as 'difficult' for fold recognition, and treated as a test set by CE and ProSup. The comparison of CLePAPS with DALI, CE and ProSup is shown in Table 1. It is difficult to compare N_e and RMSD directly for different methods. To make a close comparison with DALI, we superimpose a given structure pair according to the DALI alignment, and remove residue duads with distances greater than d_0 and the aligned segments whose lengths are smaller than ρ. The remaining reduced correspondence is the DALI-core of the original alignment. The transformation optimal to the core is then determined, and N_e and RMSD are calculated. Similarly, we also derive the CE-core alignments from the original CE alignments. Generally, alignments of CLePAPS are comparable with those of other alignment tools.

Table 1. Comparison of structure alignments obtained by DALI, CE, ProSup and CLePAPS for 10 'difficult' cases from the Fischer benchmark. N_e: total number of equivalent residue duads; rmsd: RMSD in the unit of Å; IDA: number of residue duads which are identical to those of DALI.

Pair		CE	DALI	CE-core	DALI-core	ProSup		CLePAPS	
		N_e/rmsd	N_e/rmsd	N_e/rmsd	N_e/rmsd	N_e/rmsd	IDA	N_e/rmsd	IDA
1fxiA	1ubq	64/2.8	60/2.6	59/2.5	55/2.3	54/2.6	41	55/2.4	42
1ten	3hhrB	87/1.9	86/1.9	85/1.7	84/1.7	85/1.7	79	84/1.7	77
3hlaB	2rhe	85/3.5	75/3.0	71/3.0	63/2.3	71/2.7	37	65/2.3	57
2azaA	1paz	85/2.9	81/2.5	73/2.5	76/2.1	82/2.6	8	78/2.3	72
1cewI	1molA	81/2.3	81/2.3	78/2.0	78/1.9	76/1.9	68	78/2.0	75
1cid	2rhe	98/3.0	97/3.2	79/2.0	82/2.0	84/2.3	70	87/2.2	72
1crl	1ede	220/3.9	211/3.5	155/2.5	168/2.5	161/2.6	147	169/2.7	146
2sim	1nsbA	276/3.0	292/3.3	236/2.5	240/2.5	248/2.6	231	248/2.6	213
1bgeB	2gmfA	109/4.6	94/3.3	62/2.7	79/2.2	87/2.4	0	82/2.4	0
1tie	4fgf	117/3.0	114/3.1	99/2.3	97/2.2	101/2.4	48	100/2.3	94

The CLePAPS alignment for the pair 1bgeB: 2gmfA has nothing in common with the first of DALI's three alignments. The List-20 of the protein pair has 31 members, but none coincides with any segment of the first DALI alignment. This means that the local similarity of the alignment is rather weak. The CLePAPS

alignment for the pair is very similar to the second DALI alignment of $N_e = 94$ with RMSD 3.3Å. We take five proteins of different structure classes from the benchmark as query to be aligned with each of the target structures of the benchmark. CLePAPS is able to find the related strucutres as highly similar to the queries. To be less greedy, CLePAPS generates several alignments from star centers of highly scored SFPs. Often there is one alignment which has a much higher N_e than others. There are situations where several meaningful alignments do exist when structures contain repeats, symmetry, or domain moves. Unlike algorithms using dynamic programming, CLePAPS is able to detect non-topological structural similarity of domain shuffling.

3 BLOMAPS: A Fast Multiple Structure Alignment Tool

Multiple structure alignment carries significantly more information than pairwise alignment. Most existing methods of multiple structural alignment combine a pairwise alignment and some heuristics with a progressive-type layout to merge pairwise alignments into a multiple alignment (Ye and Janardan, 2004; Guda et al., 2004; Lupyan et al., 2005). Besides the computational cost, such pairwise-based methods have the limitation that the alignments that are optimal for the whole input set might be missed. There are a handful of truly multiple methods. A way to conduct multiple alignment is to start with sets of structurally common fragments extracted from as many input proteins as possible, and then combine them into a global common substructure. For example, in doing this, MASS (Dror et al., 2003a; 2003b) implements a two-level alignment, using both secondary structure and atomic representation. we have developed a fast and reliable tool called BLOMAPS based on our conformational alphabet.

3.1 Highly Similar Fragment Blocks

The correspondence of a multiple alignment defines an equivalence relation of residues among proteins. This will be called the 'vertical equivalency'. For any two structures in the multiple alignment, the transformation to superimpose the residue duads in a subset of the correspondence will also bring the residue duads in the complement set of the correspondence spatially close. This is the 'horizontal consistency'. The requirement of both the equivalency and consistency increases the difficulty of multiple alignment, but also reduces the chance of making an irrelevant alignment. The latter gives freedom to greedy algorithms.

The extension of the concept of SFPs to multiple alignment is that of similar fragment blocks (SFBs). Selecting a structure from the input set as a template, and taking a string from its conformational sequence as a seed, by string comparison we can search for SFPs between the template and any structure in the set other than the template. An SFB may then be built up by collecting one SFP from each protein possessing members of SFPs. Usually, more than one SFP may be found between two proteins, so for a given seed many SFBs can be constructed. To avoid this 'combinatorial explosion', we introduce the concept of 'highly similar fragment blocks' (HSFBs). An HSFB of a given seed is the SFB

formed by the seed and its neighbors which score the highest amongst neighbors in each structure possessing neighbors. Of course, it could happen that a seed does not have any neighbor in some structures. The total number of fragments in an HSFB will be called the depth of the HSFB. Another characteristic of an HSFB is its total score Σ which is the sum of σ scores. We define the consensus letter of a set of conformational letters as the letter which belongs to the set and has the highest sum of CLESUM scores between itself and all letters in the set. The consensus of an HSFB is then defined as the string which consists of the consensus letters of columns when we align the 'row' strings of the HSFB. Thus, besides the positions of its member fragments, an HSFB has a width, depth, score and consensus. Our greedy algorithm uses the shortest protein to create HSFBs with width $l = 12$ and $T_{12} = 200$. To remove redundancy we sort the HSFBs first in descending order of depth and then of score, then keep only the HSFBs which have limited overlap with the already existing HSFBs.

3.2 Scaffold Building-Up

BLOMAPS starts with an HSFB taken from the top five HSFBs as an 'anchor'. To choose the most representative structure, we update the template to the protein whose member in the anchor HSFB is closest to the block consensus. Since HSFBs are created using the shortest protein it may happen that the updated template, or the new pivot protein, does not have a fragment in an HSFB. In this case we use the consensus of the HSFB to search the pivot protein for possible neighbor fragments, and add the optimal neighbor to the HSFB. All structures will be then aligned against this pivot protein. After we have superimposed all the structures which have a fragment in the anchor HSFB against the template, we examine the horizontal consistency by examining the separations of SFPs in HSFBs with cutoff $d_1 = 12\mathring{A}$. We mask inconsistent SFPs from HSFBs. If an HSFB finally contains less than 3 fragments it will be regarded as inconsistent; if a fragment of the anchor HSFB is supported by less than two SFPs it will be removed from the anchor HSFB. From the top five anchor HSFBs, we select the optimal one to go on to the next step by inspecting first the total number of consistent HSFBs, and then the total number of consistent fragments if necessary. The optimal anchor HSFB is then best supported both horizontally and vertically. Fragments of consistent HSFBs on the template form a scaffold for multiple alignment.

3.3 Obtaining the Final Alignment

Improving the scaffold. So far the transformation to superimpose two structures is based on a single SFP. Using the consistent fragments, we may update the transformation. With the transformation updated, we use a width $l' = 8$ and a more stringent cutoff $d_2 = 8\,\mathring{A}$ to examine the consistency and add fragments by a procedure of 'recruiting aligned fragment pairs (AFPs)' for every anchored protein. We sort SFPs in descending order of scores, examine the separations of the SFPs in succession to expand alignment. We then obtain an extended scaffold. The AFPs map residues of proteins other than the pivot to those of the pivot protein, and define columns of residue correspondence. We construct

the first average template by averaging transformed coordinates over atoms in individual columns.

Dealing with unanchored structures. It may happen that the anchor HSFB has no consistent neighbor in some structures. No transformation can be found based on the anchor HSFB to superimpose such an 'unanchored' protein on the pivot protein. However, the protein may still have members in other consistent HSFBs. Any of such members can be used to generate a transformation for superimposing the protein on the template, and then examining the consistency of other fragments. If an unanchored protein does not have enough number of consistent fragments, we have to try to align it pairwise on the template. Being different from the usual pairwise alignment, the scaffold on the pivot protein now provides a guide. A procedure similar to, but simpler than, CLePAPS can be used. After having succeeded in superimposing unanchored proteins on the template, we update the average template, and then examine the deviation between residue duads of AFPs and their flanking sites. We may elongate or shrink fragments according to the deviation cutoff $d_3 = 5\,\mathring{A}$, and hence update the AFPs. The modified AFPs lead to an updated average template. This is an iteration, and its convergence is usually rather fast.

Missing motifs. Due to the greedy nature of the above approach, only patterns shared by the pivot protein have a chance to be discovered. Some patterns could be shared by a subset of structures, but be absent from the pivot protein. They are 'missing motifs' to the pivot protein. Their information has to be extracted from the structures sharing them. A motif must be an SFB. There are many methods for discovering motifs in a set of sequences. We may use a simple center-star approach in the sense of strings (Zheng, 2005). An exceptionally large protein will have a large proportion of blank regions, most of which have no contribution to missing motifs, which makes the center-star approach rather inefficient. This inefficiency occurs also when the number of structures is large. We propose another way to rescue missing motifs. We divide the space occupied by the structures after superimposition into uniform cubic cells of a finite size, say 6Å. The number of different proteins which have their residues falling in a given cell is the depth of the cell. We discard cells of low depths, then sort the remaining cells in descending order of depth. Picking a cell of a large depth as a base, in each dimension we select from its two neighboring cells the one with the higher depth to expand the base cell and double its size into an 'octad'. A fragment falling in the octad may be taken as a seed to search for motifs.

In the case when a common core is shared by only a subset of the input set, we divide the latter into two subsets: one with that core and the other without it. The algorithm first accomplishes the alignment for the first subset, and then treats the second subset as a new input set.

Evaluation. The final alignment is given by the complete residue correspondence. A full column of the correspondence has residues from every protein of the structure set. The common core of the alignment is rigorously defined by all full columns. We may also define a 'partial core' by introducing a parameter of

proportion. For example, core-60 is given by columns covering 60% or more of the proteins of the structure set. Assume that a new motif is found on a protein other than the pivot. With the help of the common core, we can map the protein together with the missing motif on the template, and update the template by averaging coordinates over residues in individual new columns. After having superimposed structures on the final average template, we can calculate the total squared deviation of aligned residues with respect to the template for full or a partial core of the alignment, and then derive the RMSD for evaluation.

3.4 Test of BLOMAPS

BLOMAPS has been tested on 17 protein structure ensembles. These ensembles, covering various challenging cases of structural alignment, are taken from several references. Some ensembles contain structural homologies at different levels, some exhibit submotifs, symmetry, repetition, or different topologies, while others contain a large number of proteins. The ensembles are briefly summarized in Table 2. BLOMAPS starts by taking the shortest portein as a pivot for finding HSFBs. If the shortest one poorly represents the set, BLOMAPS will get

Table 2. Comparison of BLOMAPS alignments with those of CE-MC and MAMMOTH-mult on 17 ensembles. Size: ensemble size, L: length range; MicrRib: Microbial ribonucleases, Subtil: Subtilisins, TIM61: a set of 61 TIM barrels, Serin5: a set of 5 Serine proteinases, Serpin: Serpins, Thior: Thioredoxins, Beta: All beta immunoglobulins, Glob10: a set of 10 Globins, Glob16: another set of 16 Globins, Serin68: another set of 68 Serine proteinases, CaBind: Calcium-binding proteins, CL-GL: Cofilin-like/Gelsolin-like proteins, PLP: PLP-dependent transferases, C2: C2-domains, AlpBet: SCOP α/β, TIM7: another set of 7 TIM barrels, HelBun: Helix-Bundles. [d]: Dror et al. (2003a), [e]: Dror et al. (2003b), [s]: Shatsky et al. (2002), [y]: Ye and Janardan (2004).

			BLOMAPS		MAMMOTH		CE-MC	
Name	Size	L	$N_{1.5}$	N_c	N_c^*	$N_0+N_\pm$	N_c	$N_0+N_\pm$
MicrRib[e]	63	100:104	99	99	–	–	–	–
Subtil[e]	60	263:281	256	257	–	–	–	–
TIM61[e]	61	384:443	337	342	–	–	–	–
Serin5[s]	5	274:279	216	249	223	214+ 0	242	224+ 0
Serpin[s]	13	337:420	265	301	226	220+ 1	305	289+ 6
Thior[y]	10	85:112	67	70	43	38+ 1	79	63+ 6
Beta[y]	6	95:115	58	77	56	53+ 2	79	60+11
Glob10[y]	10	136:158	97	114	91	89+ 0	114	104+ 2
Glob16[y]	16	136:158	87	101	46	45+ 0	95	89+ 3
Serin68[d]	68	181:396	108	110	–	–	–	–
CaBind[s]	6	75:185	40	49	0	0+ 0	58	25+ 0
CL-GL[e]	12	96:174	40	63	40	38+ 0	67	56+ 3
PLP[d]	11	361:730	77	142	–	–	211	116+18
C2[d]	10	123:841	61	66	–	–	71	53+ 4
AlpBet[y]	4	81:226	28	44	0	0+ 0	58	0+26
TIM7[s]	7	247:491	17	97	0	0+ 0	79	0+ 0
HelBun[s]	10	79:159	8	59	0	0+ 0	44	0+ 0

warned at the very beginning, and a second protein has to be taken as a new pivot. However, this does not happen for all the 17 ensembles.

Since various criteria are used it is difficult to define a general comparison between different aligning methods. A simple way is to look for some identity rate between the compatible cores of alignments of two methods. Here we take the BLOMAPS alignment as a reference. Corresponding columns from two cores of alignment are recognized by counting their identical site indices. When we compare BLOMAPS with CE-MC (Guda et al., 2004), the identical indices are summed, after divided by the ensemble size, to give the effective number of 'identical columns' N_0. To include also small shifts, we count site indices which deviate at most two sites in the related columns, and the number of counts is converted to another effective number of columns $N_{\pm}$. The number $N_0 + N_{\pm}$ may be then compared with the total length N_c of the common core of alignment. MAMMOTH-mult provides a 'strict core' of alignment which is shared by all members in an ensemble and has the distance deviation between any aligned residue duad within $4.0\mathring{A}$ after superimposition (Lupyan et al., 2005). The total length of this strict core is denoted by N_c^*. At a column RMSD cutoff of $1.5\mathring{A}$, we obtain a stringent core of BLOMAPS, whose size is denoted by $N_{1.5}$ to compare with N_c^*. We summarize the comparison of BLOMAPS with MAMMOTH-mult and CE-MC also in Table 2. The MASS alignment is not as easy to be compared as MAMMOTH-mult and CE-MC, and is then not included in the table. It is seen that alignments of BLOMAPS generally agree with those of CE-MC or MAMMOTH-mult. Ensembles CaBind, AlpBet, TIM7 and HelBun contain members from different superfamilies or even different folds, besides the symmetry and repetition. Thus, observing some discrepancy among different methods in these ensembles is not so surprising. It should be mentioned that N_c of a common core of CE-MC alignment and N_c of BLOMAPS are not directly comparable. For example, the value of $N_c = 58$ is found by CE-MC for ensemble CaBind although its $N_{1.5}$ is only 16.

4 Discussion

CLePAPS and BLOMAPS distinguish themselves from other existing algorithms for structure alignment in the use of conformational letters. The description of 3D segmental structural states by a few conformational letters aptly balances precision with simplicity. The substitution matrix CLESUM provides us with a proper measure of the similarity between these discrete states or letters. Such a description fits the ϵ-congruent problem very well. Furthermore, CLESUM extracted from the database FSSP of structure alignments contains information of structure database statistics. For example, although two frequent helical states are geometrically very similar, scores between them are relatively low, which reduces the chance of accidental matching of two irrelevant helices. The conversion of coordinates of a 3D structure to its conformational codes requires little computation. Once we transform 3D structures to 1D sequences of letters, tools for analyzing ordinary sequences can be applied with some modification. The

use of conformational letters for fast local similarity search can be integrated into many existing tools to improve their efficiency.

The CLESUM similarity score can be used to sort the importance of SFPs and SFBs for greedy algorithms. Guided by CLESUM scores, only the top few SFPs and HSFBs need to be examined to determine the superposition for alignment, and hence a reliable greedy strategy becomes possible. Since many computational steps are conducted on conformational codes instead of 3D coordinates, and the search space is dramatically reduced by sorting, they run much faster than other tools. The running time for the 68 pairs of the Fischer benchmark is less than 2 percent of that of the downloaded CE local version. The longest run time among the tested 17 ensembles is spent for TIM61: 3.7 s for BLOMAPS vs. 2879 s for MASS.

References

Dror, O., Benyamini, H., Nussinov, R., Wolfson, H.: MASS: Multiple structural alignment by secondary structures. Bioinformatics 19(suppl. 1), 95–104 (2003a)

Dror, O., Benyamini, H., Nussinov, R., Wolfson, H.: Multiple structural alignment by secondary structures: Algorithm and applications. Protein Science 12, 2492–2507 (2003b)

Fischer, D., Elofsson, A., Rice, D., Eisenberg, D.: Assessing the performance of fold recognition methods by means of a comprehensive benchmark. In: Proc. Pac. Symp. Biocomput., pp. 300–318 (1996)

Gibrat, J.F., Madej, T., Bryant, S.H.: Surprising similarities in structure comparison. Current Opinion in Structural Biology 6, 377–385 (1996)

Guda, C., Lu, S., Sheeff, E.D., Bourne, P.E., Shindyalov, I.N.: CE-MC: A multiple protein structure alignment server. Nucleic Acids Res. 32, W100–W103 (2004)

Holm, L., Sander, C.: Dali/FSSP classification of three-dimensional protein folds. Nucleic Acid Res. 25, 231–234 (1997)

Kabsch, W.: A discussion of the solution for the best rotation to related two sets of vectors. Acta. Crystal. 34A, 827–828 (1978)

Lackner, P., Koppensteiner, W.A., Sippl, M.J., Domingues, F.S.: ProSup: A refined tool for protein structure alignment. Protein Engineering 13, 745–752 (2000)

Lupyan, D., Leo-Macias, A., Ortiz, A.R.: A new progressive-iterative algorithm for multiple structure alignment. Bioinformatics 21, 3255–3263 (2005)

Shatsky, M., Nussinov, R., Wolfson, H.: MultiProt – A multiple protein structural alignment algorithm. In: Guigó, R., Gusfield, D. (eds.) WABI 2002. LNCS, vol. 2452, pp. 235–250. Springer, Heidelberg (2002)

Shindyalov, I.N., Bourne, P.E.: Protein structure alignment by incremental combinatorial extension (CE) of the optimal path. Protein Engineering 11, 739–747 (1998)

Vriend, G., Sander, C.: Detection of common three-dimensional substructures in proteins. Proteins 11, 52–58 (1991)

Ye, J., Janardan, R.: Approximate Multiple Protein Structure Alignment Using the Sum-of-Pairs Distance. J. Comput. Biol. 11, 986–1000 (2004)

Zheng, W.M.: Relation between weight matrix and substitution matrix: Motif search by similarity. Bioinformatics 21, 938–943 (2005)

Zheng, W.M., Liu, X.: A protein structural alphabet and its substitution matrix CLESUM. In: Priami, C., Zelikovsky, A. (eds.) Transactions on Computational Systems Biology II. LNCS (LNBI), vol. 3680, pp. 59–67. Springer, Heidelberg (2005), http://arxiv.org/abs/q-bio/0412046

On-the-Fly Rotamer Pair Energy Evaluation in Protein Design

Andrew Leaver-Fay[1], Jack Snoeyink[2], and Brian Kuhlman[1]

[1] Department of Biochemistry, University of North Carolina at Chapel Hill
[2] Department of Computer Science, University of North Carolina at Chapel Hill

Abstract. Most existing algorithms for protein design, including those in the Rosetta molecular modeling program, precompute energies for rotamer pairs, since these energies can be examined repeatedly. Simulated annealing algorithms, however, do not examine these energies with the same frequency; while some are examined many times, others may not be examined at all. This paper compares strategies for computing these energies on the fly and caching computed energy values that are likely to be reused. By avoiding the expense of computing pair energies that are not examined by simulated annealing, we show that some caching strategies not only improve running time in design, but also use 90% less memory, which allows design computations to be performed on memory-limited machines.

1 Introduction

Protein design programs typically precompute interaction energies between pairs of rotamers from a rotamer library [1,2]. Since the number of these energies scales quadratically with the number of possible rotamers per residue, and since recent work has shown that increasing the sampling of rotamers per residue increases the likelihood of finding a good design [3,4], memory becomes a principal limitation for design. Although the computer science concept of "virtual memory" allows a processor to work on data that exceeds the available memory, the excess is swapped out to disk, whose access times are typically 100–1000 times slower than physical memory. To avoid this slowdown, designers must perform their computations on computers with large amounts of physical memory, restricting the pool of machines that can perform their design simulations.

The simple solution to the memory problem is to eliminate energy storage by replacing energy look-up with energy calculation. Unfortunately, energy calculation is often 100–1000 (or more) times slower than energy lookup: if an optimization algorithm requests the same interaction energy multiple times, then relying on on-the-fly calculations costs time. Thus, one might expect this paper to be a simple exploration of the usual trade-off: speed for memory. Instead, this paper demonstrates a performance gain in tandem with a memory-use reduction by exploiting a property of simulated annealing optimization algorithms: they need not examine all possible pair energies.

I. Măndoiu, R. Sunderraman, and A. Zelikovsky (Eds.): ISBRA 2008, LNBI 4983, pp. 343–354, 2008.

1.1 Previous Work

The task of the optimization algorithm task is to solve the *sidechain placement problem* and optimize rotamer placement on a fixed protein backbone:

Sidechain Placement Problem: Given a rotamer library, an energy function, a fixed protein backbone, and a selected set of residues on that backbone, find the assignment of rotamers to these selected residues that minimizes the energy function.

If the given energy function is pairwise decomposable, then this problem can be expressed as a state assignment in a graph, $G = \{V, E\}$, in which each vertex corresponds to residue, and each edge corresponds to the interaction between rotamers assigned to a pair of residues positions. The rotamer choice at a selected residue is expressed as a state assignment to the corresponding vertex $v \in V$, and the interaction energy between a chosen rotamer and the background is captured as a one-body energy, $\mathcal{E}_v(S_v)$, that is a function of the state S_v assigned to v. The interaction energies between a pair of rotamers at vertices u and v are captured as a two-body interaction function, $\mathcal{E}_{u,v}$, assigned to edge (u, v). The side chain placement problem becomes finding the state assignment vector, S, that minimizes

$$\sum_{v \in V} \mathcal{E}_v(S_v) + \sum_{\{u,v\} \in E} \mathcal{E}_{u,v}(S_u, S_v).$$

The side chain placement problem is NP-Complete [5], and brute force optimization to assign rotamers to n residues, with s rotamers per residue, would require $\Theta(s^n)$ time. Exact techniques for solving this problem have appealed to many protein designers; these techniques include dead-end elimination [6,7,8,9,10], branch and bound [11,12], and dynamic programming [13,14]. None of these techniques, though, can escape the inherent complexity of the problem, and each can fail to produce an answer in a reasonable amount of time.

Stochastic techniques have proven an attractive alternative since they always finish, even if they do not guarantee optimality. Such techniques include self-consistent mean field [15], genetic algorithms [16], simulated annealing [17,18,19], and recently, the FASTER method [20,21].

Most design software relies on precomputing and tabulating rotamer pair energies, since most optimization algorithms use each rotamer pair energy repeatedly. We use asymptotic analysis to give an idea of table size and number of energy examinations, even though the specific implementation determines the constants. For this analysis and the rest of the paper, we assume that we are working with a short-ranged energy function. With a short ranged energy function, and a density limit on the number of residues per unit volume, the number of neighbors with which a single residue interacts is bound by a constant [22].

Assuming a short-ranged energy function, n residues being redesigned (tens), and s rotamers per residue (thousands to tens of thousands), an algorithm must tabulate $O(ns^2)$ rotamer pair energies. The original dead-end elimination theorem [6] examines $O(ns^3)$ pair energies; the Goldstein theorem [8] examines $O(ns^4)$ pair energies; the fuzzy-ended theorem for dead-end pairs [7] examines

$O(ns^7)$ pair energies; the k-th generalized theorem [10] examines $O(ns^{2k+1})$ pair energies. For the recently described FASTER algorithm, when relaxation is performed only for neighbors of the perturbed residue(s), application of a single round of sPBR to each rotamer costs $O(ns^3)$ time and application of a single round of dPBR to each rotamer pair costs $O(ns^5)$ time. For both dead end elimination and FASTER, precomputing pair energies is readily justified since its $O(ns^2)$ expense is asymptotically smaller than the number of pair-energy examinations. (If the energy function is instead assumed to be long-ranged – as it would be if a traditional Coulombic electrostatic term were included – then the exponent on n increases by at least one for all the above techniques and for the expense of precomputing and storing pair energies.)

In contrast, simulated annealing need not re-examine all rotamer pair energies. Briefly, simulated annealing considers a series of stochastic rotamer substitutions, and decides to accept or reject each substitution by applying the stochastic Metropolis criterion to the change in energy induced. This criterion depends on a "temperature parameter" that is gradually lowered bias acceptance toward lower energy assignments. The temperature schedule and the number of rotamer substitutions considered are at the discretion of the user.

The molecular modeling program Rosetta, developed for protein structure prediction [23,24] and protein design [19,25,26,27,28], relies on a simulated annealing algorithm for its side chain placement. Even though simulated annealing offers no guarantee of optimality, independent runs converge to similar energies. The simulated annealing algorithm within Rosetta considers a linear number of rotamer substitutions ($200 \times ns$), and therefore examines a linear number of rotamer pair energies – each rotamer substitution requires examination of $O(1)$ pair energies. For this reason, precomputing pair energies dominates the running time: the cost of precomputing all pair energies takes $\Theta(ns^2)$ time whereas simulated annealing takes $\Theta(ns)$ time. The performance improvement described here is from replacing $\Theta(ns^2)$ energy computations before simulated annealing with $\Theta(ns)$ energy computations during simulated annealing.

2 Methods

This paper examines three strategies for on-the-fly (OTF) rotamer pair energy evaluation, and compares them to precomputing all rotamer pair energies. The three OTF strategies differ in how much memory they dedicate to caching of those pair energies they actually compute: At one extreme, the *cacheless* strategy stores no pair energies and recomputes each pair energy as it is requested. time. At the other extreme, the *full caching* strategy caches every pair energy evaluated. And somewhere in the middle, the *partial caching* strategy stores some pair energies but not all. Later, we show that the memory use for both the cacheless and partial caching strategies scales linearly with the number of rotamers in the design problem; the differences between the two are the constants that are hidden by asymptotic analysis. The memory use of the full caching strategy

that we examined scales quadratically with the number of rotamers per residue, though it could have been implemented to scale linearly.

2.1 Energy Evaluation

For an assignment of rotamers to all positions, the sum of one-body and two-body energies is the energy for the structure. Even with OTF strategies, the one-body energies can be precomputed because they take only linear time and memory; only the two-body energies need to be computed during simulated annealing. Thus, we try to partition the energy calculations at a residue position so that interactions that do not depend upon a rotamer choice are captured by the one-body energy function.

Rosetta's full atom scoring function [29] includes four terms that are two-body interactions: a Lennard-Jones term, a pairwise decomposable implicit solvation term [30], a hydrogen bonding term [31], and a knowledge-based electrostatics term [32]. The first three terms sum atom pair interactions, and the last is by residue pair.

Since Rosetta performs fixed backbone design, it is natural to divide the atoms of a residue into backbone (bb) and side chain (sc), and to divide the interaction energies for a pair of residues, i and j, into four groups: $\mathcal{E}(bb_i, bb_j)$, $\mathcal{E}(sc_i, bb_j)$, $\mathcal{E}(bb_i, sc_j)$ and $\mathcal{E}(sc_i, sc_j)$. If we assume that the backbone does not change as a result of an amino acid or rotameric substitution (an assumption that may not hold for mutations to or from proline), then the $\mathcal{E}(bb_i, bb_j), \mathcal{E}(sc_i, bb_j)$, and $\mathcal{E}(bb_i, sc_j)$ energies may be stored in the one-body energies for rotamers i and j, leaving only $\mathcal{E}(sc_i, sc_j)$ in the two body energies. To calculate $\mathcal{E}(sc_i, sc_j)$ efficiently, we use a trie ordering of the atoms, pruning atom-pair computations based on the non-overlap of subtree bounding spheres, in much the same way that the Trie vs Trie algorithm prunes computations [33]. Proline residues force a correction term, since their backbone nitrogen has a different covalent bonding pattern than the other amino acids.

Proline Correction. Because the backbone nitrogen of proline does not form hydrogen bonds, the implicit solvation and hydrogen bonding terms in Rosetta treat proline backbone units different than the backbone units of other amino acids. Thus, if i can mutate to or from proline, the interaction energy $\mathcal{E}(bb_i, bb_j)$ is not constant and the energy $\mathcal{E}(bb_i, sc_j)$ is not a one-body energy for residue j.

One could handle this by computing bb/bb and bb/sc energies as part of the two-body energy during simulated annealing; this however proved significantly slower than computing sc/sc energies alone. Instead, when a residue i can mutate to or from proline, we include a corrective term to the one-body energy for each rotamer on i's neighbors. Since proline is a single exception, these corrective terms may be precomputed and stored using time and memory that scales linearly with the number of rotamers in the design. Without distinguishing proline's backbone from other backbones, the energies of the produced designs were 0.3 energy units worse on average.

2.2 On-the-Fly Strategies

This section describes the three strategies for evaluating the energy of a specific pair of rotamers considered by simulated annealing optimization. As described above, we can think of this as assigning states to vertices of a graph in which vertices hold one-body energy functions and edges hold two-body energy functions. Because Rosetta's energy function is short ranged, and because there is a density limit for residues, the number of neighbors each residue has is limited by a constant [22], and the corresponding graph representing their interactions is sparse. The strategies differ by whether they associate memory with these graph edges to cache values from energy function evaluations. In contrast, Rosetta currently uses a precompute-and-store strategy that evaluates the energy function for each edge and stores the value for lookup by the optimizer.

Cacheless. The cacheless strategy is a true on-the-fly (OTF) evaluation for pair energies: Every time the graph is asked for the change in energy induced by substituting rotamer s on residue i with a new rotamer, s', it (re-)computes the interaction energies of s' with each rotamer assigned to the neighbors of i.

The memory use for the cacheless graph is $\Theta(ns)$; each edge stores proline corrections and $\Theta(1)$ auxiliary data, each vertex stores rotamer coordinates and $\Theta(1)$ auxiliary data. Because the edges do not store any pair energies after they are computed, the cacheless graph repeats rotamer pair energy computations between some rotamer pairs; over the course of $\Theta(ns)$ rotamer substitutions, it computes $\Theta(ns)$ rotamer pair energies. Asymptotically, the cacheless strategy is preferable to the precompute-and-store strategy for Rosetta's simulated annealing optimizer.

Full Caching. The full caching strategy for OTF evaluation of pair energies stores each rotamer pair energy after it is computed. Each time a pair energy is required, the graph looks to see if it has already computed that energy – if it has not, then it computes the energy and stores the energy for later reuse. This technique ensures that each rotamer pair energy is computed at most once.

One implementation option for full caching that we did not explore is to store a hash table on the graph edges. This option would store $O(ns)$ pair energies after $O(ns)$ rotamer substitutions, though we expect the expense of storing the keys to the hash table to mute its memory efficiency, and the time to index into the hash table to mute its time savings. Such an implementation may nonetheless prove interesting.

The implementation option presented in this paper is to allocate the full $\Theta(s^2)$ energy table on each edge, and then to mark each element in the table with a non-physical sentinel value (-1234). During a rotamer substitution, the graph retrieves a set of values from the edge tables; it then compares each retrieved value against the sentinel. If a retrieved value equals the sentinel, then the graph computes the corresponding rotamer pair energy and replaces the sentinel in the pair energy table. Otherwise, the retrieved value is the correct rotamer pair energy.

The memory use for this second implementation is $\Theta(ns^2)$ – as much memory as the precompute-and-store strategy. The running time for this strategy has an asymptotic bound of $\Theta(ns^2)$, since $\Theta(ns^2)$ memory must be allocated and initialized to the sentinel. However, memory allocation and initialization is fast and the actual running time for this strategy is dominated by the pair energy calculations during simulated annealing. Here, the full caching strategy computes $O(ns)$ rotamer pair energies over the course of $\Theta(ns)$ rotamer substitutions while ensuring that no pair energy evaluation is ever repeated.

Partial Caching. The partial caching strategy for OTF pair energy evaluation is to store some rotamer pair energies after they are computed, but to eventually discard those stored energies. The key to deciding which pair energies to keep and which to discard is in the recognition that in mid- to late-stage simulated annealing, when the temperature is low, most rotamer substitutions are rejected. If most rotamer substitutions are rejected, then the currently assigned rotamer r on residue i will stay assigned to i for an extended period of time. If neighbor j is asked to consider the substitution to state s, the interaction energy between r and s must be computed. If the substitution is rejected, then neighbor j may be asked to consider the substitution to state s again in the future. If at that time, residue i is still assigned rotamer r, then the interaction energy between r and s will again be needed. The key is to store interaction energies with the currently assigned rotamers since the interaction energies with those rotamers are most likely to be needed again in the future.

Indeed, saving the interaction energies with the currently assigned rotamers can be extended to saving the interaction energies for the k-most recently assigned rotamers for some constant k. Intuitively, if a rotamer r was assigned to residue i and then replaced by rotamer r', then the rotamer assignment to i's neighbors must make r look good, even if not as good as for r'. It remains likely that i might be substituted back to state r in the near future, at which time interaction energies with rotamer r will again be needed.

The partial caching strategy is as follows. Each vertex tracks its k most recently assigned states. The edge between vertices i and j stores all computed interaction energies between any rotamer on vertex i with the k most recently assigned states on vertex j as well as all computed interaction energies between any rotamer on vertex j with the k most recently assigned states on vertex i. Whenever vertex i is assigned a new state that is not among its k most recently assigned states, each edge incident upon i discards the stored pair energies for the $(k+1)$st most recently assigned state on i.

The edge between vertices i and j stores two tables; if there are s_i states for vertex i and s_j states for vertex j, then edge $\{i, j\}$ stores one table of size $k \times s_i$ and a second table of size $k \times s_j$. The edge tracks which energies it has already computed by setting entries in these tables for energies it has not computed to a sentinel value. When state r on residue i joins the set of the k most recently assigned states to i, then the edge $\{i, j\}$ sets all of the energies in a single row of the $k \times s_j$ table to the sentinel value – it wipes clean the values stored in that row which correspond to the $(k+1)^{st}$ most recently assigned state on residue i.

A single row in this table can be wiped in amortized constant time if each edge also stores two tables (of size $k \times s_i$ and $k \times s_j$) to track which energies it has already computed. Alternatively, the edge can wipe the whole row in $\Theta(s)$ time.

The expense of the $\Theta(s)$ wipe is most noticeable at high temperatures where rotamer substitutions are often accepted. At high temperature, the amount of time a rotamer is likely to remain assigned is short – the benefit of storing energies for later reuse is slight, and the cost of wiping clean energies stored in edges is large. For this reason, the partial caching graph behaves like the cacheless graph when the frequency of rotamer-substitution acceptances is high. To measure the acceptance frequency, each vertex keeps track of the acceptance/rejection status of its last 100 considered rotamer substitutions and informs its incident edges to track energies for its k most recently assigned states only if 5 or fewer of the last 100 rotamer substitutions were accepted. Maintenance of the acceptance/rejection history can be performed in constant time. With this acceptance-frequency-sensitive behavior switch for the partial caching graph, we have found that the $\Theta(s)$ wipe is faster than the amortized $\Theta(1)$ wipe for all values of ns that we have examined (up to 200K).

The memory use for this partial caching graph scales as $\Theta(kns)$, where we typically use a k of 10. The running time in simulated annealing for $\Theta(ns)$ rotamer substitutions is $O(ns)$ for the amortized constant-time wiping scheme, and $O(ns^2)$ for the linear-time wiping scheme.

3 Results

We compared running times for the three on-the-fly strategies against the running times for the precompute strategy at complete-protein-redesign tasks of twelve globular, single-chain proteins. The design tasks designed all residues using all amino acids except cysteine. Disulfides, however, were kept fixed. For each protein, we also examined five different schemes, defined in Table 1, for producing rotamers by additional sampling of torsion angles χ_1 and χ_2. After each rotamer was built, its interaction energy with the backbone was calculated; rotamers colliding the with background were discarded. The number of rotamers that passed the backbone-collision test and the amount of space required to store one floating point number for each interacting rotamer pair are reported in Table 2. All design calculations were performed on 2.8 GHz Xeon processors with 4 GB of RAM. Table 3 contains the running times.

On a per-RPE (Rotamer Pair Energy) basis, it is faster to use the trie-vs-trie algorithm [33] to compute all RPEs for a pair of interacting residues than it is to compute each RPE individually. For this reason, the precompute strategy is faster than any of the OTF strategies when the number of rotamers in the design problem is small (as in A), as nearly all RPEs are examined during simulated annealing. For rotamer sampling schemes B through E, however, the full caching strategy is the fastest. For strategies C though E, the partial caching strategy is faster than the precompute strategy in almost all cases. Indeed the partial caching strategy computes less than one fifth of all possible rotamer pair energies

Table 1. Five rotamer sampling schemes for our study. Scheme A took rotamer samples from the centers of the rotamer distributions reported by Dunbrack [1]. The other schemes expanded the set of allowed rotamers for buried residues by also considering rotamers placed one standard deviation (σ) away from the distribution centers for either χ_1 or χ_2 dihedrals. Schemes D and E also sample at one-half of a standard deviation from the distribution center. Scheme C is commonly used by the protein design community. Schemes D and E have been prohibitively expensive until now.

Sampling Scheme	Extra χ_1 Samples	Extra χ_2 Samples	Samples per starting rotamer
A	–	–	1
B	$\pm\ \sigma$	–	3
C	$\pm\ \sigma$	$\pm\ \sigma$	9
D	$\pm\{0.5, 1\}\ \sigma$	$\pm\ \sigma$	15
E	$\pm\{0.5, 1\}\ \sigma$	$\pm\{0.5, 1\}\ \sigma$	25

when the number of rotamers exceeds 100K. The cacheless strategy performed substantially worse (> 3 times slower) than the partial caching strategy in all design tasks (data not shown).

To examine the effect of the recent history size on performance, we examined the behavior of the partial caching strategy for recent history sizes of 1, 2, 3, 4, 5, 7, 10, 15, 20, 25, and 30. We performed design simulations on the same set of twelve proteins using the same set of rotamer sampling schemes. In each design simulation, we counted the number of RPEs that the annealer examined, and the number of RPEs the partial caching strategy had to compute – that is the number of RPEs that were not already stored in memory at the time the annealer requested them. Figure 1 shows that the average fraction of requested RPEs that were computed decreases in response to increasing recent history sizes. The fractions are the average across all twelve proteins for each rotamer sampling scheme and each recent history size. The figure shows diminishing returns as memory is exchanged for speed; recent history sizes much larger than 10 are only marginally faster.

4 Discussion

Protein design remains a computationally demanding task; while the side chain placement problem on a fixed backbone may be broached in a reasonable amount of time using stochastic techniques, the number of possible backbone conformations is vast. In the *de novo* design of novel protein backbones, for example, it is unknown whether a hypothetical backbone scaffold is designable until several rounds of sequence/structure optimization and gradient based minimization have completed. Hundreds of thousands of hypothetical scaffolds are pushed through the design algorithm so that a handful of designed sequences can be selected for synthesis and experimental characterization. Expediting the design process allows the computational biochemist to explore a larger region of backbone conformational space before investing several months on the bench synthesizing their designs.

Table 2. Rotamer counts (thousands) and Rotamer Pair Energy (RPE) memory usage. The rotamer counts exclude all rotamers that collide with the backbone. The memory-use projection includes space to store RPEs between only those rotamer pairs within an amino-acid-pair specific Cβ distance cutoff. These cutoffs reflect the short range nature of Rosetta's energy function. The actual memory use for the partial caching strategy is based on a recent-history size of 10 and excludes the expense of storing the rotamers. The average memory savings of the partial caching scheme in comparison with the precompute and full caching schemes is A) 62%, B) 81%, C) 92%, D) 95%, and E) 97%.

	# Rotamers (K)					MB Needed for All RPEs					Partial Caching, MB				
PDB	A	B	C	D	E	A	B	C	D	E	A	B	C	D	E
1bx7	5	6	8	11	15	10	18	53	107	214	4	5	8	11	15
1a8o	8	9	14	18	25	28	49	148	305	628	11	14	23	32	44
1g2b	8	12	24	37	55	37	117	602	1486	3494	13	21	44	68	101
1ail	9	12	22	33	47	39	93	408	964	2231	13	18	35	52	77
1aba	11	16	32	48	72	51	156	824	2054	4908	18	29	61	94	141
1bkf	12	21	47	74	111	60	240	1483	3855	9338	24	43	98	154	234
1d4t	13	21	44	68	102	61	224	1337	3434	8403	23	40	90	140	212
1aac	13	23	52	82	125	68	297	1930	5058	12507	26	49	115	182	278
1erv	13	23	54	84	129	71	315	2060	5406	13471	25	48	114	180	278
1bkr	13	23	52	82	124	64	263	1667	4345	10600	26	48	112	177	269
1a62	15	26	59	93	142	77	321	2045	5325	13162	29	54	126	200	306
1bkb	17	27	57	88	132	85	292	1650	4173	10070	30	50	110	170	258

In this paper, we have compared performance and memory use of three on-the-fly energy-calculation- and caching-strategies – new to the domain of protein design – against the standard practice of precomputing and storing all energies. The full-caching strategy uses as much memory as the precompute strategy, but is 2.6 times faster for typical design problems. Importantly, the partial-caching strategy has proven itself slightly faster than precomputing all pair energies while using dramatically less memory. Designers may rely on this technique when memory limits would otherwise prevent them from harnessing a particular machine, without having to sacrifice speed.

Recent trends in commodity processors show an increase in the number of processors per machine. Graphics processors lie at the furthest extreme of this trend but remain out of reach for the protein design community currently. A more modest goal would be to fully harness the now commonplace dual-core and quad-core machines. However, the amount of RAM has not grown proportionally to the number of processors – in effect, available memory is shrinking. The caching technique described in this paper will allow designers to run many independent jobs on multi-core machines without overflowing into virtual memory.

The reduced memory overhead for the partial caching strategy has enabled us to perform millions of *de novo* protein-protein interface designs on an IBM BlueGene/L computer. The particular BlueGene has four-thousand processors, but each processor has only 512 MB of RAM and no virtual memory. Harnessing its power required the retooling of our software as described in this paper.

Table 3. Running times for the full caching, precomputed, and partial caching strategies, in seconds. Timings include all stages of design; from the start of rotamer creation to the conclusion of simulated annealing. Bold numbers are minimum in their category. Cells marked with '–' reflect design tasks that crashed after requesting too much memory. For rotamer sampling scheme C, the full caching strategy is 2.6 times faster on average than the precomputed scheme.

	Full Caching					Precomputed					Partial Caching, $k = 10$				
PDB	A	B	C	D	E	A	B	C	D	E	A	B	C	D	E
1bx7	16	**14**	**35**	**55**	**47**	**14**	20	38	56	127	27	35	62	94	113
1a8o	44	**58**	**64**	**158**	**220**	**40**	60	136	265	530	83	108	184	281	432
1g2b	**50**	**101**	209	**336**	–	**50**	127	443	927	–	86	149	**208**	602	**1,046**
1ail	60	88	**190**	**203**	**475**	**48**	**74**	255	449	937	114	174	358	407	622
1aba	76	137	**197**	**520**	–	**68**	**122**	599	1,485	–	135	227	339	606	**1,641**
1bkf	101	**133**	**357**	–	–	**83**	**246**	979	–	–	170	339	909	**1,622**	**2,839**
1d4t	97	**192**	**483**	–	–	**59**	244	771	–	–	173	206	538	**1,506**	**2,725**
1aac	106	**227**	**385**	–	–	**99**	**251**	1,400	–	–	179	382	671	**1,140**	**3,439**
1erv	120	**234**	**626**	–	–	**101**	314	1,237	–	–	188	249	1,088	**1,915**	**3,515**
1bkr	109	**154**	**609**	–	–	**89**	262	1,090	–	–	189	247	616	**1,935**	**3,276**
1a62	139	**263**	**443**	–	–	**109**	**334**	1,374	–	–	233	458	1,269	**2,295**	**2,496**
1bkb	132	**164**	**390**	–	–	**119**	**265**	1,320	–	–	227	273	1,127	**2,043**	**3,555**

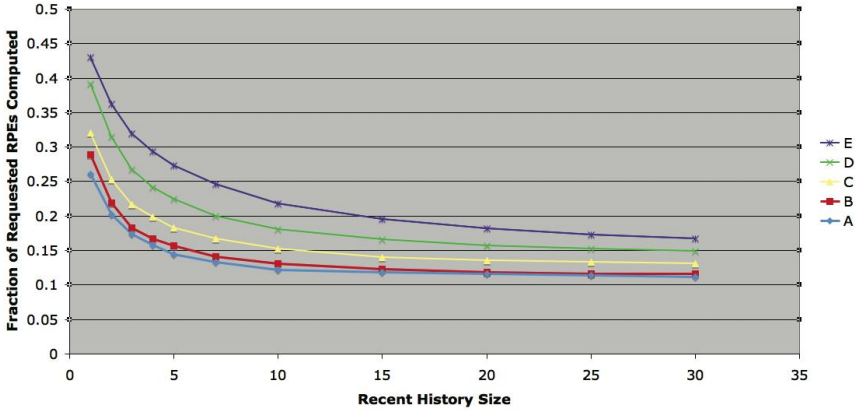


Fig. 1. Fraction of rotamer pair energies requested during simulated annealing that were computed (as opposed to those RPEs that had already been computed and were simply retrieved from memory). By reducing the recent history size from 30 to 10, the number of RPEs computed increases only by 9 %(A), 13% (B), 16% (C), 21% (D), and 31% (E).

Acknowledgments

We would like to thank the Renaissance Computing Institute (RENCI) for access to their BlueGene machine which in part lead toward the development of the

algorithms described within. We would also like to thank the reviewers for their helpful comments and criticisms. This research was funded by DARPA's PDP initiative and from NIH Grant GM073960.

References

1. Dunbrack Jr., R.L., Karplus, M.: Backbone dependant rotamer library for proteins: Application to side chain prediction. Journal of Molecular Biology 230, 543–574 (1993)
2. Lovell, S.C., Word, J.M., Richardson, J.S., Richardson, D.C.: The penultimate rotamer library. Proteins: Structure Function and Genetics 40, 389–408 (2000)
3. Looger, L.L., Dwyer, M.A., Smith, J.J., Hellinga, H.W.: Computational design of receptor and sensor proteins with novel functions. Nature 423, 185–190 (2003)
4. Kuhlman, B., O'Neill, J.W., Kim, D.E., Zhang, K.Y., Baker, D.: Accurate computer-based design of a new backbone conformation in the second turn of protein L. Journal of Molecular Biology 315, 471–477 (2002)
5. Pierce, N., Winfree, E.: Protein design is NP-hard. Protein Engineering 15, 779–782 (2002)
6. Desmet, J., Maeyer, M.D., Hazes, B., Lasters, I.: The dead-end elimination theorem and its use in protein side-chain positioning. Nature 356, 539–541 (1992)
7. Lasters, I., Desmet, J.: The fuzzy-ended elimination theorem: Correctly implementing the side chain placement algorithm based on the dead-end elimination theorem. Protein Engineering 6, 717–722 (1993)
8. Goldstein, R.F.: Efficient rotamer elimination applied to protein side-chains and related spin glasses. Biophysical Journal 66, 1335–1340 (1994)
9. Gordon, D.B., Mayo, S.L.: Radical performance enhancements for combinatorial optimization algorithms based on the dead-end elimination theorem. Journal of Computational Chemistry 19, 1505–1514 (1998)
10. Looger, L.L., Hellinga, H.W.: Generalized dead-end elimination algorithms make large-scale protein side-chain structure prediction tractable: Implications for protein design and structural genomics. Journal of Molecular Biology 307(1), 429–445 (2001)
11. Gordon, D., Mayo, S.: Branch-and-terminate: A combinatorial optimization algorithm for protein design. Structure Fold Des 7, 1089–1098 (1999)
12. Canutescu, A.A., Shelenkov, A.A., Dunbrack Jr., R.: A graph-theory algorithm for rapid protein side-chain prediction. Protein Science 12, 2001–2014 (2003)
13. Leaver-Fay, A., Liu, Y., Snoeyink, J.: Faster placement of hydrogen atoms in protein structures by dynamic programming. In: 6th Workshop on Algorithm Engineering and Experiments (ALENEX 2004) (2004)
14. Leaver-Fay, A., Kuhlman, B., Snoeyink, J.: An adaptive dynamic programming algorithm for the side chain placement problem. In: Pacific Symposium on Biocomputing, The Big Island, HI, pp. 17–28. World Scientific, Singapore (2005)
15. Koehl, P., Delarue, M.: Application of a self-consistent mean field theory to predict protein side-chains conformation and estimate their conformational entropy. J. Mol. Biol. 239(2), 249–275 (1994)
16. Desjarlais, J., Handle, T.: *De novo* design of hydrophobic cores of proteins. Protein Science 4, 2006–2018 (1995)
17. Holm, L., Sander, C.: Fast and simple Monte Carlo algorithm for side chain optimization in proteins: Application to model building by homology. Proteins 14(2), 213–223 (1992)

18. Hellinga, H., Richards, F.: Optimal sequence selection in proteins of known structure by simulated evolution. Proceedings of the National Academy of Sciences, USA 91, 5803–5807 (1994)
19. Kuhlman, B., Baker, D.: Native protein sequences are close to optimal for their structures. Proceedings of the National Academy of Sciences, USA 97, 10383–10388 (2000)
20. Desmet, J., Spriet, J., Lasters, I.: Fast and accurate side-chain topology and energy refinement (FASTER) as a new method for protein structure optimization. Proteins 48, 31–43 (2002)
21. Allen, B.D., Mayo, S.L.: Dramatic performance enhancements for the faster optimization algorithm. Journal of Computational Chemistry 27, 1071–1075 (2006)
22. Xu, J.: A tree-decompositon based approach to protein structure prediction. In: Miyano, S., Mesirov, J., Kasif, S., Istrail, S., Pevzner, P.A., Waterman, M. (eds.) RECOMB 2005. LNCS (LNBI), vol. 3500, pp. 423–439. Springer, Heidelberg (2005)
23. Simons, K.T., Bonneau, R., Ruczinski, I., Baker, D.: Ab initio protein structure prediction of CASP III targets using ROSETTA. Proteins: Structure Function and Genetics 37, 171–176 (1999)
24. Bradley, P., Chivian, D., Meiler, J., Misura, K., Rohl, C., Schief, W., Wedemeyer, W., Schueler-Furman, O., Murphy, P., Strauss, J.S.C., Baker, D.: Rosetta predictions in CASP5: Successes, failures, and prospects for complete automation. Proteins: Structure Function and Genetics 53, 457–468 (2003)
25. Kuhlman, B., Dantas, G., Ireton, G., Varani, G., Stoddard, B., Baker, D.: Design of a novel globular protein fold with atomic-level accuracy. Science 302, 1364–1368 (2003)
26. Dantas, G., Kuhlman, B., Callender, D., Wong, M., Baker, D.: A large scale test of computational protein design: Folding and stability of nine completely redesigned globular proteins. Journal of Molecular Biology 332, 449–460 (2003)
27. Ashworth, J., Havranek, J., Duarte, C., Sussman, D., Monnat, R.J., Monnat, R.J., BL, B.S., Baker, D.: Computational redesign of endonuclease dna binding and cleavage specificity. Nature 441, 656–659 (2006)
28. Sood, V., Baker, D.: Recapitulation and design of protein binding peptide structures and sequences. Journal of Molecular Biology 357, 917–927 (2006)
29. Rohl, C., Strauss, C., Misura, K., Baker, D.: Protein structure prediction using rosetta. Methods in Enzymology 383, 66–93 (2004)
30. Lazaridis, T., Karplus, M.: Effective energy function for proteins in solution. Proteins: Structure Function and Genetics 35, 133–152 (1999)
31. Kortemme, T., Morozov, A.V., Baker, D.: An orientation-dependent hydrogen bonding potential improves prediction of specificity and structure for proteins and protein-protein complexes. Journal of Molecular Biology 326, 1239–1259 (2003)
32. Simons, K., Ruczinski, I., Kooperberg, C., Fox, B., Bystroff, C., Bystroff, C., D., D.B.: Improved recognition of native-like protein structures using a combination of sequence-dependent and sequence-independent features of proteins. Proteins: Structure Function and Genetics 34, 82–95 (1999)
33. Leaver-Fay, A., Kuhlman, B., Snoeyink, J.: Rotamer-pair energy calculations using a trie data structure. In: Casadio, R., Myers, G. (eds.) WABI 2005. LNCS (LNBI), vol. 3692, pp. 500–511. Springer, Heidelberg (2005)

Invited Keynote Talk: Integrative Viral Molecular Epidemiology: Hepatitis C Virus Modeling

James Lara, Zoya Dimitrova, and Yuri Khudyakov

Molecular Epidemiology and Bioinformatics Laboratory, Division of Viral Hepatitis, Centers for Disease Control and Prevention, 1600 Clifton Road, Atlanta, GA, 30333, USA
{jlara,zdimitrova,ykhudyakov}@cdc.gov

Abstract. Traditional molecular epidemiology of viral infections is based on identifying genetic markers to assist in epidemiological investigation. The limitations of early molecular technologies led to preponderance of analytical methodology focused on the viral agent itself. Computational analysis was almost exclusively used for phylogenetic inference. Embracing the approaches and achievements of the traditional molecular epidemiology, integrative molecular epidemiology of viral infections expands into a comprehensive analysis of all factors involved into defining outcomes of exposure of a person(s) to viral infections. The major emphasis of this scientific discipline is on the development of predictive models that can be used in different clinical and public health settings. The current paper briefly reviews a few examples that illustrate a new trend in integrative molecular epidemiology striving to quantitatively define viral properties and parameters using primary structure of viral genomes.

1 Introduction

Significant advances in development of molecular approaches made over the last two decades rendered studies on structural variations in genetic systems of various organisms not only conceivable but clearly affordable. These developments spurred an avalanche of applications in different fields but almost none benefited as much as epidemiologic research for public health. The application of molecular approaches to the study, prevention and control of health risks in human populations is frequently described as molecular epidemiology [1]. However, such description of molecular epidemiology reduces it to the application of molecular markers to epidemiological research. The latest progress in molecular and computational technologies challenges such description. Molecular epidemiology was recently defined as "a science that focuses on the contribution of potential genetic and environmental risk factors, identified at the molecular level, to the etiology, distribution and prevention of disease within families and across populations" (http://www.pitt.edu/~kkr/molepi.html).

Application of novel molecular and computational approaches to viral research transforms molecular epidemiology of viral infections into an integrative research

I. Măndoiu, R. Sunderraman, and A. Zelikovsky (Eds.): ISBRA 2008, LNBI 4983, pp. 355–366, 2008.

discipline that embraces studies of viral and host genetic systems, molecular evolution, phylogeny and population genetics and sets these studies in specific epidemiological situations. Although primarily a biological discipline, integrative molecular epidemiology is deep-rooted into mathematical and computational modeling. Qualitative or modestly accurate quantitative mathematical models are important tools for guiding molecular epidemiological research, while sufficiently accurate quantitative or predictive models can be directly used in clinical and public health settings. Currently, molecular research is still dominated by analytical approaches which attempt to consider all components of biological systems separately from each other. Recent advances in computational and high throughput molecular technologies afford a rather comprehensive analysis of all components involved in determining the outcome of exposure of a person or population of people to certain risk factors, which sets the foundation for integrative molecular epidemiology. The important issue to consider is that viral molecular epidemiology is interested in genetic heterogeneity within species, should it be a host or viral agent. At this level, many environmental parameters and circumstances of exposure and infection defined by different epidemiological settings are as important to consider as biological properties of the pathogen or host. This consideration poses a novel challenge to molecular epidemiological research of viral infections to quantitatively assess the effect of sequence heterogeneity of a viral genome on its functions, which is frequently defined as a quantitative structure-activity relationship (QSAR) problem in computational studies. Among many important trends in the integrative molecular epidemiology of viral infections, this paper will briefly review a few QSAR models developed in our laboratory for hepatitis C virus (HCV).

HCV is the major etiologic agent of blood borne non-A, non-B hepatitis. HCV genome is a positive-sense single-stranded 9.6 kb RNA. The HCV genome encodes a large polyprotein, which is processed into 10 mature proteins. The genome is very heterogeneous. Some HCV strains may differ as much as 35% from each other. Phylogenetic analysis of HCV sequences identified 6 genotypes. Clinical studies have shown that HCV genotype is an independent prognostic factor in predicting response to antiviral therapy [2]. Although it is well documented that sequence heterogeneity of HCV genome affects antigenic properties of encoded proteins [3], the degree of genotype specific differences in antigenic properties is not known. Due to the characteristic geographical distribution of all HCV genotypes worldwide, current diagnostic kits do not always perform equally well in all parts of the world. Therefore, the development of affordable HCV diagnostic assays with improved specificity and sensitivity continues to be a major public health challenge.

2 HCV NS3 Cross-Immunoreactivity

One of the strategies adopted for generating immunoreactive forms of HCV antigens for their use in diagnostic assays involves characterization of antigenic determinants derived from different HCV strains. However, probing of the entire HCV sequence space for the existence of molecules with desired properties is too onerous to be

practicable. An alternative approach is to define QSAR between protein structure and antigenicity for the development of antigens with improved diagnostic properties.

The HCV NS3 protein contains diagnostically relevant conformation dependent immunodominant B cell epitopes. One of the HCV NS3 conformational antigenic regions could be efficiently modeled with recombinant proteins of 103 amino-acids (aa) long [4]. The effect of sequence variation on antigenic property of this region was studied using a set of 12 recombinant proteins derived from 6 known HCV genotypes. The study showed that some changes in primary structure can result in a significant variation of antigenic properties.

2.1 Dataset

Twelve HCV NS3 protein variants comprising the amino-acid positions 331 – 433 of the HCV NS3 helicase domain or positions 1357 – 1459 of the HCV polyprotein have been expressed using synthetic genes and tested by enzyme immunoassay (EIA) against a panel of anti-HCV positive sera of patients from diverse geographical settings and infected with different HCV genotypes [4]. HCV NS3 sequences were encoded using several physicochemical property scales: hydrophobicity [5], volume [6] and polarity [6]; secondary structure information [7, 8]; and the first 3 or 5 eigenvector components derived by principal component analysis from a collection of 143 amino-acid properties [6]. Variants were tested against 115 anti-HCV positive serum specimens. Of these, 107 serum samples were included in the training data set, as eight samples did not bind to any of the synthetic NS3 proteins. The strength of serum reaction to the NS3 variants was measured as EIA S/Co values. For ANN training, S/Co values were normalized to range from 0 to 1. Also, irrelevant proteins (polyglycine and poly-alanine), and randomized sequences with equal amino-acid composition to NS3 variants were included in the training set to provide the ANN with negative examples.

2.2 ANN Model

An ANN model capable of predicting the antigenic properties of HCV NS3 proteins was developed. The ANN architecture used in this work is the fully connected feed forward network consisting of three layers of neurons. The number of input units in the input layer was set according to the input vector dimensions (103 to 618 inputs), and the number of output units in the output layer was set to 107 (the number of serum samples in the training set). The final number of hidden units in the hidden layer was set to 159 units based on accuracy of simulations. The learning algorithm used to train the ANN was back propagation with momentum [9], and the generalized delta rule [10] was used as the cost function for updating the weights for error minimization. ANN performance was measured by Leave-one-out cross-validation (LOOCV).

The best overall performance from ANN simulations was observed using the physicochemical scales of normalized hydrophobicity, volume and polarity (accuracy of 89%). To further validate our ANN model, contribution analysis of individual protein

positions to antigenic properties was performed by evaluating the relative weights of the connections between the first and hidden layer of the neural network from each physicochemical attribute along the sequence strand. Positions associated with large relative weights in the ANN were mapped on the HCV NS3 protein structure (PDB: 1cu1). Mapped residues (total of 25 residues) grouped into three major clusters. Additional predictions for localization of antigenic regions on the surface of the HCV NS3 protein were obtained using the CEP server [11, 12] and antigenic prediction server [13]. The strong concordance between these 3 clusters and predicted antigenic determinants validates the relevance of associations identified with the ANN model.

The described ANN model allowed for a rapid in silico testing of a large number of HCV NS3 sequences of different genotypes collected from GenBank. Figure 1 shows that NS3 proteins of different genotypes demonstrate a broad range in predicted breadth of immunoreactivity. HCV NS3 protein variants from genotype 1a/1b were all immunoreactive with <65% of serum specimens. The variants from genotype 2 were all predicted to be broadly immunoreactive (>70% of serum specimens). Significant disparity in distribution of antigenic properties among different HCV subgenotypes suggests significant functional differences between subgenotypes and should be taken into consideration during diagnostic, molecular virological and molecular epidemiological research. The HCV NS3 proteins derived from subgenotypes 1c, 2a and 2b are among the most suitable targets for assay development.

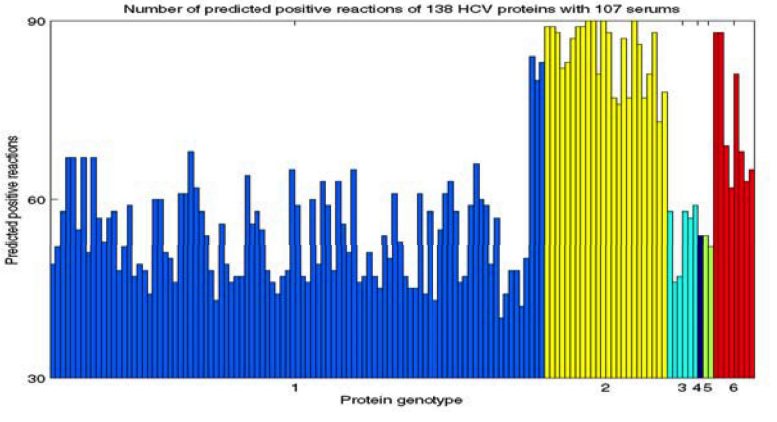


Fig. 1. Predicted antigenic reactivity of 138 HCV NS3 sequences with anti-HCV positive sera. y axis: number of positive reactions (107 samples); x axis: genotype.

Due to the labor intensive nature of experimental quantitative evaluation of engineered proteins, the data sets generated by these experiments are frequently limited in size. This poses a serious challenge to a reliable mathematical modeling of QSAR. The ANN model built for this study is most probably overfitted. However, results from the HCV NS3 3-D structure mappings, suggests that the described ANN model is completely suitable for guiding research or a focused rational design of antigenic targets with improved diagnostically relevant properties through a cyclic process of

experimental evaluation of predicted antigens and re-training the model for a more accurate representation of QSAR in this specific case of the HCV NS3 conformational antigenic epitope.

3 HCV HVR1 Cross-Immunoreactivity

The ANN model described above allowed for discovering an important functional disparity between HCV genotypes and subtypes. This model suggests that some subtypes in their entirety are inferior targets for antibody binding. Although a very important observation, it relates only to the antibody response against a nonstructural protein. HCV nonstructural proteins, however, do not elicit any neutralizing humoral responses. A known neutralizing epitope(s) is located at the N-terminus of the HCV envelope E2 protein, which due to its significant heterogeneity is known as hypervariable region 1 (HVR1). This region contains only 27 aa. Currently, a prevailing hypothesis is that a significant variability of the neutralizing antigenic epitope(s) located within HVR1 is the major molecular mechanism responsible for the HCV evasion of neutralization by antibodies. Therefore, it is crucial to understand quantitative relationship between HVR1 structure and immunological properties. Such QSAR model can be used to guide designing novel vaccine candidates and to study cross-immunoreactivity of different HCV strains in the course of molecular epidemiological investigation.

3.1 Dataset

In our study, 5 different HCV HVR1 variants, with amino-acid identities ranging from 37.0% to 59.3%, were engineered into the N-terminus of the hepatitis B virus surface antigen (HBsAg). All 5 HVR1-HBsAg recombinant proteins were expressed in yeast in the form of virus like particles (VLP). After purification, these VLP's were used to immunize mice in order to obtain anti-HVR1 specific antibody. Sera obtained from mice immunized with these proteins were tested against a panel of 172 synthetic peptides representing major HCV HVR1 variants. Because of a very weak immunoreactivity of sera against 3 constructs, only data obtained from testing antibodies against HC3 and HC5 HVR1-HBsAg VLP's were used for QSAR modeling.

3.2 SVM Model

Support Vector Machine (SVM) was used to identify positions and structural parameters that determine anti-HVR1 antibody cross-immunoreactivity. Classification analysis was conducted to examine the difference between the broadly cross-immunoreactive HVR1 peptides and peptides showing limited, if any, cross-immunoreactivity. The results were used to explore the association between the HVR1 structural parameters and cross-immunoreactivity. We have identified 2 classes among all immunoreactive HVR1 peptides used in this study. Class 1 consists of 74 peptides that immunoreacted only with sera from mice immunized with HC3 HVR1-HBsAg construct (n=14) or immunoreacted only with sera from mice immunized with HC5

HVR1-HBsAg construct (n=60). This class defines differences in immunoresponses to HC3 and HC5 HVR1-HBsAg constructs. Class 2 consists of 22 peptides that immunoreacted with sera from mice immunized with HC3 and sera from mice immunized with HC5 constructs. This class defines shared immunoreactivity or cross-immunoreactivity for antibodies obtained against HC3 and HC5 HVR1-HBsAg constructs. Peptides that did not immunoreact with any of the sera were not considered.

To determine the best peptide representation, sequences were scanned using a sliding window of length 5, 7 and 9. For each window hydropathy, isoelectric point [14] and flexibility [15]were calculated. Three different hydropathy tables were used for the representations as described by Hopp [16], Kyte [17] and Eisenberg [18].

SVM*light* implementation of SVM with polynomial kernel of degree 2 achieved the best classification accuracy when a sequence representation using sliding window of size 7 and Hopp's hydropathy table, isoelectric point and flexibility calculated for each window was used. Using this model, we have conducted experiments for selection of peptide properties. The properties whose removal could have improved the classification accuracy were excluded from the data set. When removal of more than one property leads to the same improvement of the classification rate, the property to be removed was selected randomly from the set of the candidates. It was shown that the overall *One vs. All* accuracy calculated as an average for three runs with property selection was 92%, the accuracy for class 1 (HC3 or HC5 specific immunoresponses) was 95%, and for class 2 (shared immunoreactivity for HC3 and HC5) was 83%. Hydropathy and flexibility were the most important properties for accuracy of the classifaction.

3.3 BN Model

Using same data set as described above, we have developed also Bayesian Network (BN) model for the identification of association between the HCV HVR1 structure and cross-immunoreactivity. As for SVM, class 1 and class 2 peptide data were used for training BN using the same attributes.

Deriving the BN classifier consists of two tasks. The first task is to select the BN structure (i.e. finding the dependency structure that achieves the highest score). The K2 algorithm [19] was implemented for learning causal structure. In order to decrease the level of complexity, structural constraints were added to the algorithm so as to restrict the number of parents that each node could have to three. In addition, formation of output links from the class node to any other variables of the network was restricted. The second task is parameter estimation, which for a directed graph consisted of specifying the conditional probability tables (CPT) at each node. In this case, direct estimates of the conditional probabilities were obtained by maximum likelihood estimates. Accuracy of the classification was assessed by 10-fold Cross-Validation (CV).

The best BN classifier based on the 5-bin discretization of the Hopp's hydropathy, isoelectric point, and flexibility parameters calculated for a sliding window of 7 had an accuracy of 98%. Subsequent analysis of this BN revealed that hydropathy and isoelectric point were the most important attributes for the classification. In fact, a BN trained on these 8 attributes had the same correctness of classification as the one

trained with all attributes. BN's trained on the amino-acid nominal representation had lower accuracies of prediction of ~75% when trained by position.

The models using properties of individual amino-acid positions in the HVR1 peptides did not identify strong associations between the properties and cross-immunoreactivity. Strong associations were only obtained when peptide sequences were scanned using a sliding window of variable size for calculating such properties. Both SVM and BN modeled the association between structural parameters (calculated for a window) and cross-immunoreactivity (as defined by class 1 and 2 of peptide immunoreactivity data) with greater than 90% accuracy. It was shown that hydropathy and flexibility contribute most significantly in the association. Thus, physicochemical properties of the HCV HVR1 peptides are: (1) strongly associated with the specificity of immunoreactivity; and (2) HVR1 cross-immunoreactivity can be predicted using such properties with high accuracy. Similar models can be used for guiding the design of broadly immunoreactive HVR1 variants and for the identification of cross-immunoreactivity of HVR1 derived from different HCV strains in the course of molecular epidemiological investigation.

4 Catalytic Efficiency of HCV NS3/4 Protease Variants

Analysis of the association between enzymatic activity of viral proteins and variability of their primary structure is important for accurate measuring of viral fitness and is emerging as a very effective tool in molecular epidemiological investigations. The whole area of research is still in its infancy. However, a substantial number of enzymatic QSAR models have been developed for HIV protease and polymerase as it relates to drug response [20]. There are no such QSAR models for HCV specific enzymes and data related to this problem are practically unavailable.

4.1 Dataset

Data used for this study was obtained from Franco et al. (2007) [21]. Briefly, experimental analysis of HCV NS3/4 protease activity was conducted among HCV quasispecies identified in 3 HCV-infected individuals (A, B, and C). All 3 HCV strains belonged to genotype 1b. Individuals A and B were co-infected with HIV type 1 (HIV-1), whereas individual C was HCV mono-infected. Blood samples were obtained from three patients. Individuals A and B were treated for 48 weeks with PEG-Interferon α – 2b in combination with ribavirin. Individual A showed a sustained response to therapy, whereas individual B did not respond to therapy. A sample obtained from individual C was collected during acute episode of HCV infection; afterward this individual resolved HCV infection. A total of 296 NS3/4 protease quasispecies sequences (180 aa long) were recovered from these 3 HCV-infected patients. 109 sequences were unique. All 109 HCV NS3/4 quasispecies variants were tested for their catalytic efficiency by comparing the growth of lambda phages containing the HCV NS3/4 protease recognition site coexpressed with HCV NS3/4 protease variants with the growth of the phage coexpressed with the wild-type master protease (I389/NS3-3 protease; 100%). Catalytic activity of each was measured relative (%) to this master protease. Values ranged from 0% to 200%.

4.2 ANN Model

Sequences were transformed into numerical vectors by representing each amino-acid through its physicochemical properties of hydrophobicity [5], polarity [6] and volume [6]. Numerical values of the relative catalytic efficiency of protease quasispecies were normalized between 0 and 1. This dataset was used for regression or function approximation modeling. A dataset for classification was generated using a binary representation of catalytic efficiency. In this case variants were divided into 2 classes: active or non-active, relative to the master protease. Also, irrelevant proteins (poly-glycine and poly-alanine) and randomized sequences with equal amino-acid composition to NS3/4 protease variants were included in the training. For evaluation of the ANN classifier, a dataset with randomly assigned labels ("active" or "non-active") to the quasispecies variants was used.

The ANN classifier used in this work was the fully connected feed-forward network consisting of two layers of neurons. The number of input units in the input layer was set according to the input vector dimensions (540 inputs; 3 properties x 180 aa positions), the hidden layer varied from 0 to 20 hidden neurons and the number of output units in the output layer was set to 1 (the number of classes in the training set). The ANN was trained by using the back-propagation with momentum [9] as learning algorithm and the generalized delta rule [10] as the cost function for updating the weights of the connections. For function approximation a fully connected feed-forward ANN consisting of three layers of neurons (a multiple-layered perceptron or MLP) was used. In this case, the hidden layer was varied from 0 to 200 hidden neurons. The ANN was trained to approximate the relative catalytic activity values of protease NS3/4 protein variants based on their sequence structure using 90% of the data and evaluated with the remaining 10%.

The ANN built as MLP with function approximation was found un-trainable for purposes of developing a regression model relating variants structure with catalytic activity by the present encoding scheme. Increasing the complexity of the model (number of hidden units from 0 to 200) had no improvement on the error (SSE $\geq$ 0.37). However, when we used a classification scheme (binary encoding) we obtained a trainable ANN. The best solution on this dataset was achieved with a simple perceptron. The accuracy of the linear ANN classifier was 72.3%. Adding hidden neurons (up to 20) did not decrease the error (SSE = 0.02). This suggests that the dataset used for this study is linearly separable. This has been observed in other protease datasets as well (exemplified by the HIV-1 protease cleavage site specificity) [22]. Therefore, it may be a misuse of non-linear classifiers to apply them to this problem. The linear ANN model is simple and may provide a straightforward way to extract rules that regulate the catalytic activity of HCV NS3/4 protease. In the case where variants were randomly assigned labels, the accuracy of the ANN model predictions was < 55%. The purpose of such randomization was to test if the ANN is learning the structure in the patterns of the data, as opposed to learning the structure of random patterns in the data. A high prediction accuracy on randomized data would have indicated that the ANN was learning to explain noise rather than specific patterns.

This ANN model is a useful tool for the identification of functionally inactive variants in the mixture of quasispecies found in individual patients and may help evaluate replicative fitness of HCV strains.

5 HCV Quasispecies and Viral Load

A major challenge for molecular epidemiological research is establishing integral connections between ever growing complex sets of epidemiological, virological and molecular data associated with viral infections. HCV infection presents an especially challenging task because of diversity of the genome, many routes of transmission, variety of clinical outcomes, worldwide distribution and large human population affected by this infection. Traditional statistical approaches allow for the identification of important trends in the data but fail to provide mathematical models capable of predicting various features of infections and, therefore, are not suitable for evaluation of integral dependence between all features of infections. We have developed a set of Bayesian network models that established connections between epidemiological, demographic, virological, and molecular features associated with HCV infection [23]. In this review we would like to comment on one study that was undertaken to develop refined models of HCV-infection in the form of BN specifically targeting dependencies between molecular and virological features in a manner that is both integrative and intuitive.

5.1 Dataset

The National Health and Nutrition Examination Survey III (NHANES III), conducted during 1988–1994, provides estimates of the seroprevalence of specific enteric and sexually transmitted diseases from a nationally representative sample in the US population by various demographic, socioeconomic and behavioral characteristics (http://www.cdc.gov/nchs/about/major/nhanes/nh3data.htm). Serum specimens from 89 HCV-infected patients identified through NHANES III were used for characterizing HCV quasispecies (QS) by means of HCV HVR1 sequencing, estimation of HCV titer and identification of HCV genotype. The data comprised 1,065 HVR1 sequences (each 29 aa long), and associated information consisting of: serum viral titer (VT), genotype (GT), and number of QS variants (QSv).

5.2 BN Model

All attributes collected from the NHANES III study were discretized. A BN was used to learn the structure of the data (conditional independence and dependence between features) in an unsupervised fashion. Structure learning was performed using the PC algorithm [24]. Learning of the conditional probability distributions was done using the EM algorithm [25].

The learned BN for our data (figure 2) shows that 20 out 29 HVR1 positions are interdependent and linked to virological variables (VT, GT and QSv). HVR1 positions 1, 3 and 15 have strong direct links to QSv, positions 8, 25 and 27 to genotype, and positions 8 and 9 plus genotype to VT. Independence graphs constructed from the BN models show that these variables have strong marginal dependencies. The links between them had marginal p-values <1E-20.

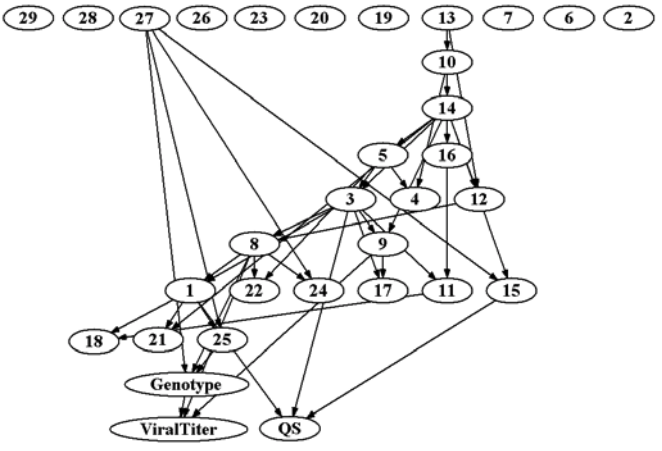


Fig. 2. Parent ordering layout of the BN model showing dependencies between HVR1 structure and virological features of HCV-infection

The predictive value of the relations between HVR1 positions to HCV serum VT, QSv and GT, as learned by the BN models, was tested by classification modeling as a way to validate the structure of our BN models. Model evaluations were performed by 10-fold CV. After extensive evaluation of the performance of the BN models, we found that based on HVR1 sequence profile alone, BN models were able to predict serum VT, QSv and GT with accuracies of 99%, 96%, and 99%, respectively.

Virological features of HCV infection presented herein were found to be strongly dependent to specific HVR1 positions. This QSAR model indicates that strain heterogeneity as measured by QSv and viral replication fitness as measured by VT are heritable traits which can be identified with a rather satisfactory accuracy using a single short region of the HCV genome. The interesting fact is that host factors were not taken into account. It seems that the HCV genome itself plays a leading role in defining QSv and VT. Such virological parameter as VT should have a strong effect on HCV transmission rate. Thus, the BN model suggests that different HCV strains have different propensity for transmission. QSv is one of important parameters that can affect HCV evolvability since it defines the size of sequence space occupied by a strain. Thus, evolutionary potential of a strain may be quantitatively estimated using QSAR models similar to described in this paper. It is conceivable that the accuracy of this model can be improved by including parameters defining epidemiologic settings such as host gender, age and race, host genetics and immunological competence, length of infection and mode of transmission.

6 Conclusion

Quantitative modeling of various aspects of viral infections is at the core of integrative molecular epidemiology. Unfortunately, there are not many examples of such modeling in viral molecular epidemiological research. The HIV research field offers a few extraordinary interesting models developed for evaluation of drug resistance

[20, 26], coreceptor usage [27-29] and replication fitness [30, 31] of different HIV sequence variants. In the field of HCV research, a few available papers describe QSAR modeling focused on virological responses to interferon treatment [32, 33]. In the present paper, we briefly reviewed HCV models developed in our laboratory for molecular epidemiological investigations. These models range from measuring enzymatic activity and immunoresponses against different HCV proteins to evaluation of virological parameters. Although the work on the HCV quantitative modeling is only at its very beginning, we believe that this field offers an exciting opportunity for computational research.

References

1. Schulte, P.A., Perera, F.A.: Molecular Epidemiology: Principles and Practice. Academic Press, London (1993)
2. Weck, K.: Molecular methods of hepatitis C genotyping. Expert. Rev. Mol. Diagn. 5, 507–520 (2005)
3. Inudoh, M., Nyunoya, H., Tanaka, T., Hijikata, M., Kato, N., Shimotohno, K.: Antigenicity of hepatitis C virus envelope proteins expressed in Chinese hamster ovary cells. Vaccine 14, 1590–1596 (1996)
4. Khudyakov, Y.E., Dou, X.-G., Chang, J., Fields, H.A.: Impact of Sequence Heterogeneity on Antigenic Properties of the Hepatitis C Virus (HCV) Proteins. Margolis, Alter, Conlon, Dienstag, Liang (eds.), pp. 381–385. International Medical Pressm, London, UK (2002)
5. Engelman, D.M., Steitz, T.A., Goldman, A.: Identifying nonpolar transbilayer helices in amino acid sequences of membrane proteins. Annu. Rev. Biophys. Biophys. Chem. 15, 321–353 (1986)
6. Schneider, G., Wrede, P.: Artificial neural networks for computer-based molecular design. Prog. Biophys. Mol. Biol. 70, 175–222 (1998)
7. Creighton, T.E.: Proteins: Structures and Molecular Properties. W.H. Freeman and Company, New York (1993)
8. White, J.V., Stultz, C.M., Smith, T.F.: Protein classification by stochastic modeling and optimal filtering of amino-acid sequences. Math. Biosci. 119, 35–75 (1994)
9. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning representations of back-propagation erros. Nature 323, 533–536 (1986)
10. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning internal representations by error propagation. In: Rumelhart, D.E., McClelland, J.L. (eds.) Parallel Distributed Processing, MIT, Cambridge (1986)
11. Kolaskar, A.S., Kulkarni-Kale, U.: Prediction of three-dimensional structure and mapping of conformational epitopes of envelope glycoprotein of Japanese encephalitis virus. Virology 261, 31–42 (1999)
12. Kulkarni-Kale, U., Bhosle, S., Kolaskar, A.S.: CEP: A conformational epitope prediction server. Nucleic Acids Res. 33, 168–171 (2005)
13. Kolaskar, A.S., Tongaonkar, P.C.: A semi-empirical method for prediction of antigenic determinants on protein antigens. Febs Lett. 276, 172–174 (1990)
14. Wilkins, M.R., Gasteiger, E., Bairoch, A., Sanchez, J.C., Williams, K.L., Appel, R.D., Hochstrasser, D.F.: Protein identification and analysis tools in the ExPASy server. Methods Mol. Biol. 112, 531–552 (1999)
15. Bhaskaran, R., Ponnuswamy, P.K.: Dynamics of amino acid residues in globular proteins. Int. J. Pept. Protein Res. 24, 180–191 (1984)

16. Hopp, T.P., Woods, K.R.: Prediction of protein antigenic determinants from amino acid sequences. Proc. Natl. Acad. Sci. U.S.A. 78, 3824–3828 (1981)
17. Kyte, J., Doolittle, R.F.: A simple method for displaying the hydropathic character of a protein. J. Mol. Biol. 157, 105–132 (1982)
18. Eisenberg, D., Schwarz, E., Komaromy, M., Wall, R.: Analysis of membrane and surface protein sequences with the hydrophobic moment plot. J. Mol. Biol. 179, 125–142 (1984)
19. Cooper, G., Herskovits, E.: A bayesian method for the induction of probabilistic networks from data. Machine Learning 9, 309–347 (1992)
20. Masso, M., Vaisman, I.I.: Accurate prediction of enzyme mutant activity based on a multibody statistical potential. Bioinformatics 23, 3155–3161 (2007)
21. Franco, S., Parera, M., Aparicio, E., Clotet, B., Martinez, M.A.: Genetic and catalytic efficiency structure of an HCV protease quasispecies. Hepatology 45, 899–910 (2007)
22. Rognvaldsson, T., You, L.: Why neural networks should not be used for HIV-1 protease cleavage site prediction. Bioinformatics 20, 1702–1709 (2004)
23. Lara, J., Gao, F.X., Xia, G., Nainan, O., Khudyakov, Y.: Bayesian networks for evaluation of dependencies between epidemiological, virological and molecular features of the hepatitis C virus infections. J. Clin. Virol. 36 (suppl. 2), pp. S119 (2006)
24. Sprites, P., Glymour, C., Scheines, R.: Causation, Prediction, and Search. MIT Press, Cambridge (2000)
25. Lauritzen, S.L.: The EM algorithm for graphical association models with missing data. Computational Statistics and Data Analysis 19, 191–201 (1995)
26. Rabinowitz, M., Myers, L., Banjevic, M., Chan, A., Sweetkind-Singer, J., Haberer, J., McCann, K., Wolkowicz, R.: Accurate prediction of HIV-1 drug response from the reverse transcriptase and protease amino acid sequences using sparse models created by convex optimization. Bioinformatics 22, 541–549 (2006)
27. Lengauer, T., Sander, O., Sierra, S., Thielen, A., Kaiser, R.: Bioinformatics prediction of HIV coreceptor usage. Nat. Biotechnol. 25, 1407–1410 (2007)
28. Sierra, S., Kaiser, R., Thielen, A., Lengauer, T.: Genotypic coreceptor analysis. Eur. J. Med. Res. 12, 453–462 (2007)
29. Skrabal, K., Low, A.J., Dong, W., Sing, T., Cheung, P.K., Mammano, F., Harrigan, P.R.: Determining human immunodeficiency virus coreceptor use in a clinical setting: Degree of correlation between two phenotypic assays and a bioinformatic model. J. Clin. Microbiol. 45, 279–284 (2007)
30. Dykes, C., Demeter, L.M.: Clinical significance of human immunodeficiency virus type 1 replication fitness. Clin. Microbiol. Rev. 20, 550–578 (2007)
31. Wu, H., Huang, Y., Dykes, C., Liu, D., Ma, J., Perelson, A.S., Demeter, L.M.: Modeling and estimation of replication fitness of human immunodeficiency virus type 1 in vitro experiments by using a growth competition assay. J. Virol. 80, 2380–2389 (2006)
32. Lin, E., Hwang, Y., Wang, S.C., Gu, Z.J., Chen, E.Y.: An artificial neural network approach to the drug efficacy of interferon treatments. Pharmacogenomics 7, 1017–1024 (2006)
33. Lin, E., Hwang, Y., Chen, E.Y.: Gene-gene and gene-environment interactions in interferon therapy for chronic hepatitis C. Pharmacogenomics 8, 1327–1335 (2007)

Multiple Kernel Support Vector Regression for siRNA Efficacy Prediction

Shibin Qiu[1] and Terran Lane[2]

[1] Pathwork Diagnostics Inc., 1196 Borregas Ave, Sunnyvale, CA, 94089, USA
sqiu@pathworkdx.com
[2] Computer Science Dept., University of New Mexico, Albuquerque, NM, 87131, USA
terran@cs.unm.edu

Abstract. The cell defense mechanism of RNA interference has applications in gene function analysis and human disease therapy. To effectively silence a target gene, it is desirable to select the initiator siRNA molecules having satisfactory silencing capabilities. Computational prediction for silencing efficacy of siRNAs can assist this screening process before using them in biological experiments. String kernel functions, which operate directly on the string objects representing siRNAs and target mRNAs, have been applied to support vector regression for the prediction and improved accuracy over numerical kernels in multidimensional vector spaces constructed from descriptors of siRNA design rules. To fully utilize information provided by string and numerical kernels, we propose to unify the two in the kernel feature space by devising a multiple kernel regression framework where a linear combination of the kernels are used. We formulate the multiple kernel learning into a quadratically constrained quadratic programming (QCQP) problem, which although yields global optimal solution, is computationally inefficient and requires a commercial solver package. We further propose three heuristics based on the principle of kernel–target alignment and predictive accuracy. Empirical results on real biological data demonstrate that multiple kernel regression can improve accuracy and decrease model complexity by reducing the number of support vectors. In addition, multiple kernel regression gives insights into the kernel combination, which, for siRNA efficacy prediction, evaluates the relative significance of the design rules.

1 Introduction

RNA interference (RNAi) is a cell defense mechanism that represses the expression of viral genes by recognizing and destroying their mRNAs, preventing them from being translated into proteins. The gene *knockdown* in RNAi is induced by short interfering RNA (siRNA) of ∼21 nucleotides (nt) long, processed from a double stranded RNA (dsRNA) by the enzyme Dicer, or transfected directly. Target mRNA transcripts that are hybridized with the siRNA are destroyed by the RNA-induced silencing complex (RISC) [1], as shown in Figure 1. RNAi has widespread applications in biology and great potentials in disease therapy [2]. Although a dsRNA of a few hundred nucleotides long is introduced into plants

I. Măndoiu, R. Sunderraman, and A. Zelikovsky (Eds.): ISBRA 2008, LNBI 4983, pp. 367–378, 2008.

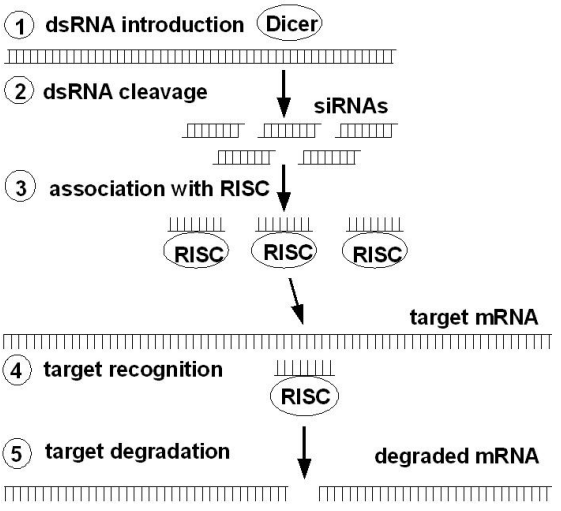


Fig. 1. The RNAi mechanism. (1) A dsRNA is introduced to initiate RNAi; (2) Dicer cleaves the dsRNA into siRNAs; (3) RISC takes up the guide strand; (4) RISC–siRNA complex recognizes the target; (5) The target mRNA is degraded by RISC.

and insects to initiate RNAi, dsRNAs longer than ∼50 bp activate the interferon (IFN) pathway in mammalian cells [3]. Therefore, siRNA molecules are directly transfected in mammals to achieve specific gene knockdown. Due to therapeutical applications, siRNA silencing efficacy becomes the focus of many biological and computational research [4].

One goal of a gene knockdown is to maximally silence the target gene. Unfortunately, if not carefully selected, an siRNA may lead to unsatisfactory or even unobservable target degradation, since siRNAs targeting different sites on the same mRNA have widely differing effectiveness and more than 70% of arbitrary siRNAs are not functional [5]. Therefore, it is crucial to predict the silencing *efficacy*, which is defined as the percentage of mRNA knockdown caused by an siRNA sequence, before using it in biological and therapeutical experiments. In this work, we develop a multiple kernel support vector learning model and apply it to the silencing efficacy prediction problem.

siRNA design rules characterizing structural and thermodynamic properties of the sequences have been proposed in the form of multidimensional numerical descriptors[5,6,7,8,9,10], which lead to input spaces for learning models [11,12]. Vector spaces can also be constructed using nucleic motifs [9] and subsequence patterns [13,14]. Sparse encoding are often used to represent the sequences in a vector space [9,15]. Using a threshold on the efficacy, 75% for example, an siRNA can be categorized into two classes: functional, if its efficacy is at least 75%, and non-functional otherwise. Then classification algorithms, such as SVM [11,13,14], decision tree [8], and neural network [10], can be applied. If an SVM is trained in a vector space, we call the kernel function (e.g Gaussian kernel) a numerical kernel as distinguished from a string kernel that operates directly on string objects.

To more accurately predict the efficacy than classification, regression methods have been used. Huesken *et al.* employed a neural network and reported prediction correlation coefficient, but error rates were not shown [9]. Moreover, neural networks are trained by gradient search, which depends on initial values and does not guarantee global optimality. Vert *et al.* used a regularized linear regression to predict the efficacy and selected significant subsequences [15]. However, this work only used linear models and did not exploit nonlinearity. Since a support vector machine employs structural risk minimization and its convex programming leads to global optimization, it has better generalization capabilities than other learning models[16]. Qiu and Lane constructed multidimensional vector spaces from descriptors of siRNA design rules to employ numerical kernels in support vector regression and achieved significant accuracies in the efficacy prediction [12,17]. They also developed and applied string kernels to the prediction and achieved higher accuracies than the numerical kernels [18,19,12,17].

Since the similarities measured by a string kernel and a numerical kernel from rule descriptors model different aspects of the siRNA samples, they contain different information and may even compliment each other. Conceivably, learning model can be improved by utilizing information from both the string and the descriptors. However, the siRNAs are presented as strings, and the descriptors, vectors in an Euclidean space. There is not a direct way to combine these two types of data. We propose a multiple kernel regression framework to unify the information in the kernel feature space, where a linear combination of string and numerical kernels are used. Multiple kernel SVM classification has been studied by Lanckriet *et al.* [20]. Since regression is generally more complicated than classification and support vector regression has a more complex formulation than its classification counterpart, a multiple kernel regression model must be developed separately. Previously, we have formulated the multiple kernel learning into a quadratically constrained quadratic programming (QCQP) problem [21]. Empirical results demonstrated that multiple kernel regression improved prediction performance and simplified model complexity by reducing the number of support vectors. Although the QCQP formulation yields global optimal solution, it is computationally inefficient and requires a commercial solver. In this work, we propose heuristics for multiple kernel learning to simplify computation.

We measure the fitness between a kernel and target labels to develop our heuristics. One of our heuristic is developed from kernel–target alignment, which was proposed for single kernel classification [22]. We derive the other two heuristics from predictive accuracy. Tests on four biological data sets demonstrate that multiple kernel regression improves predictive accuracy. Furthermore, it gives insights into the kernel combination, which provides an additional benefit of comparing the relative significance of the design rules.

2 Multiple Kernel Support Vector Regression

We summarize the support vector regression and present the multiple kernel support vector regression formulation.

Suppose we are given a set $\{(x_1, z_1), (x_2, z_2), ..., (x_l, z_l)\}$ of l training examples, where x_i $(1 \leq i \leq l)$ is a data point in an input space $\mathcal{X}$, $z_i \in \mathbb{R}$ is its target label. For siRNA sequences, $\mathcal{X}$ is a space of strings, and $z_i \in [0, 1]$ is the silencing efficacy. For siRNA design rules, $\mathcal{X}$ is an Euclidean space. We want to learn a regression function

$$f(x) = \omega^{\mathsf{T}} \phi(x) + \beta, \tag{1}$$

that can best predict the label of an unseen data point x, where ω is a weight in the kernel feature space, $\phi(x)$, the kernel feature map of x, and β, a threshold constant.

f can be solved through the following dual optimization problem [16].

$$\max_{\alpha^+, \alpha^-} -\frac{1}{2}(\alpha^+ - \alpha^-)^{\mathsf{T}} K(\alpha^+ - \alpha^-) - \varepsilon \sum_{i=1}^{l}(\alpha_i^+ + \alpha_i^-) + \sum_{i=1}^{l} z_i(\alpha_i^+ - \alpha_i^-) \tag{2}$$

$$\text{s.t.} \quad \sum_{i=1}^{l}(\alpha_i^+ - \alpha_i^-) = 0$$

$$\alpha_i^+, \alpha_i^- \in [0, C] \quad i = 1, ..., l$$

where $\alpha^+ = \{\alpha_1^+, ..., \alpha_l^+\}^{\mathsf{T}}$ and $\alpha^- = \{\alpha_1^-, ..., \alpha_l^-\}^{\mathsf{T}}$ are dual variables, and $K \in \mathbb{R}^{l \times l}$ is a kernel matrix evaluated from a kernel function $k(.,.) : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ with $K_{ij} = k(x_i, x_j)$ [23]. Parameter C adjusts the tradeoff between the regression error and the regularization on f. Solving α^+, α^-, and β using KKT (Kurash-Kuhn-Tucker) conditions applied to (2), the regression function of (1) becomes

$$f(x) = \sum_{i=1}^{l}(\alpha_i^+ - \alpha_i^-)k(x, x_i) + \beta, \tag{3}$$

where $f(x)$ depends only on training examples having nonzero coefficients (support vectors) through the representation of the kernel function k. In the regression function in (3), k is usually a single kernel function. Since a combination of kernels with nonnegative coefficients yields a legitimate kernel, we can combine them into one kernel.

Suppose we have a set $\mathcal{K}$ of kernel matrices, each of which is a linear combination of g kernel matrices generated from different kernels or the same type of kernel with different parameters. We want to learn the best combination, in addition to learning the coefficients α^+ and α^- in (3). Since kernel matrices of strong diagonal dominance overfit data and decrease generalization capability [24], we impose a constraint on the trace of the kernel matrices. Therefore,

$$\mathcal{K} = \{K | K = \textstyle\sum_{j=1}^{g} \mu_j K_j,\ K_j \succeq 0,\ \mu_j \geq 0,\ trace(K) = d\}, \tag{4}$$

where$\succeq$ denotes positive semidefiniteness, and d is some positive constant. Thus, the multiple kernel regression can be written as,

$$f(x) = \sum_{i=1}^{l}(\alpha_i^+ - \alpha_i^-)\sum_{j=1}^{g}\mu_j k_j(x_i, x) + \beta. \tag{5}$$

To proceed, we define $\gamma = \alpha^+ - \alpha^- \in \mathbb{R}^l$ and $\eta = \alpha^+ + \alpha^- \in \mathbb{R}^l$. Let e denote the vector of all ones, and $z = \{z_1, ...z_l\}^\mathsf{T} \in \mathbb{R}^l$, the target labels. The multiple kernel support vector regression can be formulated as

$$\begin{aligned} \min_{\gamma,\eta,\theta_j,v,s,t_j} \quad & v - 2z^\mathsf{T}\gamma + 2\varepsilon e^\mathsf{T}\eta \\ \text{s.t.} \quad & u = 1 \\ & at_1 - s = 0 \\ & e^\mathsf{T}\gamma = 0 \\ & -C \le \gamma \le C \\ & 0 \le \eta \le 2C \\ & t_j - t_1 = 0, \quad 2 \le j \le m \\ & 2vu \ge s^2 \\ & {H_j}^\mathsf{T}\gamma - \theta_j = 0, \quad j = 1, ..., m \\ & t_j \ge \sqrt{{\theta_j}^\mathsf{T}\theta_j}, \quad j = 1, ..., m \end{aligned} \tag{6}$$

where $a = \sqrt{2d/trace(K_j)}$ when each K_j is normalized and has trace l, and $H_j {H_j}^\mathsf{T} = K_j$ [21]. $2vu \ge s^2$ is a rotated conic constraint, and the last two constraints yields $t_j^2 \ge \gamma^\mathsf{T} K_j \gamma$, a conic constraint, resulting in a QCQP problem. Using an interior point solver, μ_j can be derived from the dual variable solution of the conic constraints. Additionally, a component kernel corresponding to a larger μ_j is more significant in characterizing the data.

It was demonstrated previously that the benefit of multiple kernel regression included accuracy increase and support vector reduction [21]. For example, a three-kernel support vector regressor reduced mean squared error (MSE) by 27.7% using 57% fewer support vectors, compared with a single kernel regressor, on the Boston housing data set [25].

Since solving (6) requires $O(gl^3)$ time, the training process can sometimes be slow, especially when the kernel matrices are dense. Instead of using the QCQP formulation, we develop easier heuristics for speedup.

3 Heuristics for Multiple Kernel Learning

Since the best prediction accuracy is generated by a best kernel combination, we quantify a fitness between a kernel and target labels to develop our heuristics.

The alignment $F(K_j, z)$ between a kernel matrix K_j and a label set z is defined as [22]

$$F(K_j, z) = \langle K_j, zz^\mathsf{T} \rangle_F / \sqrt{\langle K_j, K_j \rangle_F \langle zz^\mathsf{T}, zz^\mathsf{T} \rangle_F}, \tag{7}$$

where $\langle ., . \rangle_F$ is the Frobenius inner product, i.e. $\langle A, B \rangle_F = \sum_{i=1}^{l} \sum_{j=1}^{l} a_{ij} b_{ij}$ for matrices $A = (a_{ij}) \in \mathbb{R}^{l \times l}$ and $B = (b_{ij}) \in \mathbb{R}^{l \times l}$. Intuitively, $F(K_j, z)$ computes a similarity between two matrices similarly to a normalized inner product between

two vectors. If a kernel matrix K_j has a larger alignment to z, it should contribute a larger proportion to the combined kernel. This intuition leads to the following heuristic, which we call the F-heuristic for simplicity

$$\mu_j = F(K_j, z)/\textstyle\sum_{j=1}^{g} F(K_j, z). \tag{8}$$

Additionally, a kernel generating a larger Pearson correlation coefficient R between the predicted labels and the true labels should contribute more to the combined kernel. Let R_j be the R generated by the regressor using kernel matrix K_j. Then, our heuristic based on R, which we call the R-heuristic, is

$$\mu_j = R_j/\textstyle\sum_{j=1}^{g} R_j. \tag{9}$$

Finally, a kernel producing a smaller MSE error should contribute more to the combined kernel. Let M_j be the MSE error generated by the regressor using kernel matrix K_j. Our heuristic based on MSE (the M-heuristic) is

$$\mu_j = \frac{\sum_{i=1}^{g} M_i - M_j}{\sum_{j=1}^{g}(\sum_{i=1}^{g} M_i - M_j)} = \frac{\sum_{i=1}^{g} M_i - M_j}{(g-1)\sum_{i=1}^{g} M_i}, \tag{10}$$

where $\sum_{i=1}^{g} M_i$ is the total MSE error of all kernels, and $\sum_{i=1}^{g} M_i - M_j$ represents the contribution of K_j. For comparison of the kernel contributions, we normalize, i.e., $\sum_{j=1}^{g} \mu_j = 1$. These heuristics may not always yield global optimal solutions, but they improve prediction accuracy.

4 String Kernels and Numerical Kernels

Since we combine the RNA and the randomized string kernels (previously shown the best) with numerical kernels, we describe these kernels briefly.

The randomized string kernel. The randomized string kernel (RSK) repeatedly extracts random subsequences of length q from the input string and evaluates a similarity based on the subsequences [12]. At the i^{th} repetition, it generates an index set $I_q^i = \{j_1^i, j_2^i, ..., j_q^i\}$, where j_t^i, $1 \le t \le q$, are uniformly and uniquely drawn at random from $\{1, 2, ..., p\}$. The feature map of a string $s = \{s[1], s[2], ..., s[p]\} \in \mathcal{A}^p$ is the subsequence formed by concatenating the randomly chosen q characters,

$$\phi_{q,i}(s) = \{s[j_1^i], s[j_2^i], ..., s[j_q^i]\}. \tag{11}$$

A total of V repetitions generates a set of index sets $I_q = \{I_q^1, I_q^2, ..., I_q^V\}$, yielding the feature map $\Phi_q(s) = \{\phi_{q,1}(s), \phi_{q,2}(s), ..., \phi_{q,V}(s)\}$. RSK is defined as

$$k_q^{rsk}(s_1, s_2) = \exp\{-\gamma \textstyle\sum_{i=1}^{V} d_H(\phi_{q,i}(s_1), \phi_{q,i}(s_2))\} \tag{12}$$

where $\gamma > 0$ is a width parameter, and $d_H(.,.)$ is the Hamming distance. Due to γ, the exponent can also be viewed as the average Hamming distance.

The RNA string kernel. The RNA string kernel (RNK) simulates RNA hybridization and is defined through the notion of similarity neighborhood [19]. For $s \in \mathcal{A}^p$, its mismatch neighborhood, $N^{mis}_{m,r}(s)$, is defined as all length-p strings δ that differ from s by at most m contiguous mismatches starting at position r in s. Only contiguous mismatches are implemented (with G-U wobbles and bulges ignored), since they are frequently observed in biological experiments. The mismatch feature map of s is defined as $\Phi^{mis}_{m,r}(s) = \{\phi_\delta(s)\}_{\delta \in \mathcal{A}^p}$, where $\phi_\delta(s) = 1$ if $\delta \in N^{mis}_{m,r}(s)$, and $\phi_\delta(s) = 0$, otherwise.

For an m nt long contiguous mismatch starting at position r, the mismatched indices are $I_{m,r} = \{r, r+1, ..., r+m-1\}$, whose characters are not compared by the kernel. The substring indices that the kernel compares are in $J_{m,r} = \{1, 2, ..., p\} \backslash I_{m,r}$. We use $J_{m,r}[i]$ to denote the i^{th} element in $J_{m,r}$, $1 \le i \le |J_{m,r}|$. For each mismatch position r, s is mapped into a space of $\mathcal{A}^{p-m}$ by the following feature map, $\phi_{J_{m,r}}(s) = \{s[J_{m,r}[1]], s[J_{m,r}[2]], ..., s[J_{m,r}[p-m]]\}$.

Then the feature space induced by the mismatch kernel is spanned by all mismatch positions, $1 \le r \le p-m+1$,

$$\Phi_m(s) = \{\phi_{J_{m,1}}(s), \phi_{J_{m,2}}(s), ..., \phi_{J_{m,p-m+1}}(s)\}.$$

We compute the kernel function between two input strings s_1 and s_2 as

$$k^{rnk}_m(s_1, s_2) = \langle \Phi_m(s_1), \Phi_m(s_2) \rangle = \sum_{r=1}^{p-m+1} \langle \phi_{I(m,r)}(s_1), \phi_{I(m,r)}(s_2) \rangle_c = \sum_{r=1}^{p-m+1} \langle \widetilde{s_1}, \widetilde{s_2} \rangle_c,$$

where $\widetilde{s_1} = \phi_{J_{m,r}}(s_1)$ and $\widetilde{s_2} = \phi_{J_{m,r}}(s_2)$ are the maps of s_1 and s_2. And we compute $\langle ., . \rangle_c$ by counting the number of common characters, $\langle \widetilde{s_1}, \widetilde{s_2} \rangle_c = \sum_{i=1}^{p-m} I(\widetilde{s_1}[i] == \widetilde{s_2}[i])$, where $I(.)$ is an indicator function. Thus,

$$k^{rnk}_m(s_1, s_2) = \sum_{r=1}^{p-m+1} \sum_{i=1}^{p-m} I(\widetilde{s_1}[i] == \widetilde{s_2}[i]), \tag{13}$$

which is normalized as $\widehat{k}^{rnk}_m(s_1, s_2) = k^{rnk}_m(s_1, s_2) / \sqrt{k^{rnk}_m(s_1, s_1) k^{rnk}_m(s_2, s_2)}$.

Table 1. Summary of siRNA design rules and their encoding

Rule	Nunber of criterion	Vector space dimension	Reference
Ui-Tei	4	8	[7]
Amarzguioui	6	9	[5]
Reynolds	8	16	[6]
Jagla (first group)	4	8	[8]
Huesken's motifs	78	78	[9]

Numerical kernels from siRNA design rules. We found that in all cases, the Gaussian kernel performed the best. We therefore only need to construct the vector spaces using the siRNA design rules. For most rules, we use sparse encoding to map each siRNA sequence into a vector space. For example, the first condition of Ui-Tei's rule requires the existence of an A/U residue at the 5' end of the antisense strand. Using two binary attributes, if that position is an A

or U base, we generate a pair of (1, 0) for this criterion. Otherwise, a (0, 1). A summary is displayed in Table 1, while details omitted for space [5,6,7,8,9,17].

5 Empirical Evaluations

We first summarize the data sets. The 73 siRNA sequences targeting firefly luciferase gene and human cyclophilin B mRNA were originally used to develop Reynolds' rule [6]. We use 70 samples from this data source and call it the KR data set. Sætrom *et al.* collected 560 data points for training their SVM classifiers [11]. A small number of this data set may overlap with the KR data set. We call this data set SA data set. Jagla *et al.* tested their decision tree algorithm on 600 data points [8]. We call this data set JA data set. Huesken *et al.* studied more than 2400 siRNAs targeting 34 human and rodent mRNAs [9]. We call this 2400 siRNAs HU data set. Each data point in the data sets contains a 19-mer siRNA sequence and its corresponding efficacy score.

To use the heuristics for multiple kernel regression, we first test accuracies of single kernel regressions using numerical kernels according to the efficacy rules, the RNA and the RSK string kernels. We next combine the kernels using the weights generated by our F-heuristic, R-heuristic, and M-heuristic introduced in Section 3. We report accuracies of 10 fold cross validations on each data set.

Figure 2 (I) shows the results of multiple kernel regression using the F-heuristic. Increase in correlation coefficients is listed in Table 2. The average correlation coefficient (averaged over all data sets and all rules) using numerical kernels was 0.469. In contrast, the average R using numerical kernels combined with the RNA kernels was 0.538, representing a 14.7% increase over single-numerical kernels. On the other hand, the average R using numerical kernels combined with the RSK kernels was 0.535, representing a 14.1% increase. These increases were statistically significant, with p-values of 0.003 and 0.007, respectively. We also noticed that the standard deviations decreased when combined kernels were used. This increase in R demonstrates that combining the numerical kernel generated from an efficacy rule and an RNA string kernel or an RSK kernel using kernel alignment as weight, significantly improved prediction accuracy over using the numerical kernel alone. Meanwhile using combined kernels also yielded better accuracies than using the RSK kernel alone, although RSK performed the best among the string kernels.

In addition, results show that using combined kernels reduced the MSE error. The average MSE on all data sets using the numerical kernels was 0.0671, whereas it was 0.0614 using numerical kernels combined with the RNA string kernels, representing a decrease of 8.5%. The average MSE was 0.0622 using rule based numerical kernels combined with the RSK kernel, a decrease of 7.3%. Meanwhile, numerical kernels combined with string kernels decreased MSE errors over using RNA and RSK string kernels alone. The improvements on MSE errors are not shown in detail.

Using the R-heuristic, the average R using numerical kernels combined with the RNA kernels was 0.537, a 14.7% increase over single-numerical kernels, as shown in Figure 2 (II) and Table 2. Additionally, the average R on all data sets using numerical kernels combined with the RSK kernels was 0.533, a 13.9%

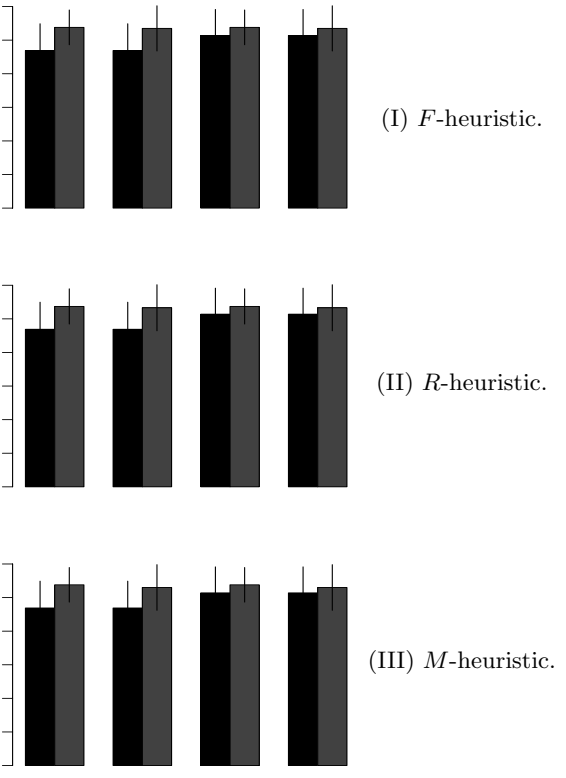


Fig. 2. Comparison of average R over the data sets of single kernel (blank bar) and multiple kernel (filled bar) regression. Vertical line represents standard deviation; "A, C", R of numerical kernel of rules; "E, G", RSK kernel; "B, F", numerical kernel combined with RNA kernel; "D, H", numerical kernel combined with RSK kernel.

increase. These increases were statistically significant, with p-values of 0.003 and 0.01, respectively. Meanwhile using combined kernels also yielded better accuracies than using the RSK kernel alone. Furthermore, results show that using combined kernels reduced the MSE and standard deviations.

Using the M-heuristic, the average R using numerical kernels combined with RNK was 0.538, achieving a 14.7% increase over single kernel regressions (Figure 2 (I) and Table 2). Also, the average R using numerical kernels combined with RSK was 0.530, a 13.0% increase. These increases were statistically significant, with p-values of 0.002 and 0.013 respectively. Furthermore, using combined kernels also yielded better accuracies than using the RSK alone. Finally, combined kernels reduced the MSE error and standard deviations.

Which design rule is better? When performing multikernel regression using the heuristics, μ_1 represents the contributions of the numerical kernels corresponding to the design rules, and μ_2, string kernels. We rank the rules according

Table 2. Percentage increase of R by multiple kernel regression. "A–H" indicate types of kernel as explained in Figure 2. "p" denotes t-test p-value.

Single vs. multiple kernel	F-heuristic	R-heuristic	M-heuristic
Rules(A) vs. Rules+RNA(B)	14.7% (p: 0.003)	14.7% (p: 0.003)	14.7% (p: 0.002)
Rules(C) vs. Rules+RSK(D)	14.1% (p: 0.007)	13.9% (p: 0.01)	13.0% (p: 0.013)
RSK(E) vs. Rules+RNA(F)	4.7%	4.5%	4.7%
RSK(G) vs. Rules+RSK(H)	4.1%	3.7%	3.1%
Average	9.4%	9.2%	8.9%

Table 3. Comparison of design rule contributions

	$\overline{\mu_1}$				
	F-heuristic	R-heuristic	M-heuristic	$\overline{\overline{\mu_1}}$	Order
Amarzguioui	0.4970	0.4834	0.5163	0.4988	2
Huesken	0.5001	0.5075	0.5124	0.5067	1
Jagla	0.4984	0.4733	0.4914	0.4877	3
Reynolds	0.4994	0.4003	0.4591	0.4529	5
Ui-Tei	0.4909	0.4684	0.4855	0.4816	4

to $\overline{\overline{\mu_1}}$, average of $\overline{\mu_1}$ (over heuristics), which is the average of μ_1 over data sets (Table 3). This ranking suggests that Huesken's motif is most significant, followed by Amarzguioui's rule, Jagla's first rule, Ui-Tei's rule, and Reynolds' rule.

6 Summary and Discussion

The siRNA molecules are intrinsically sequences, which are natively represented as strings. The design rules, on the other hand, extract numerical descriptors from the sequences. Previously, we demonstrated that string kernels achieved better prediction accuracies than design rules. To further improve prediction performance, we proposed to use multiple kernel support vector regression by combining the numerical kernels and the string kernels. Our QCQP formulation achieved optimal solutions and achieved better accuracies. But it is slow in training. For computational efficiency, we introduced three heuristics. In a typical case, the heuristic using kernel alignment as weights was more than 200 times faster than the QCQP formulation [21]. We experimented our heuristic framework on the siRNA efficacy data sets.

We found combining numerical kernels computed from efficacy rules and string kernels significantly increased prediction performances over using the numerical kernels alone. Using the heuristics, the improvements on the correlation coefficient were around 14% over the numerical kernels. Based on the performance enhancement, we found that the heuristics based on kernel alignment and on correlation coefficient gave rise to best combining weights.

According to the contribution to the combined kernel made by a numerical kernel generated by an efficacy design rule, we found that the best two rules

are Huesken's motifs and Amarzguioui's rule. Huesken's motifs are based on statistically significant single nucleotide motifs corresponding to efficient and inefficient siRNA sequences, and cover a wide range of important occurrences. Amarzguioui's rule also covers many descriptions for siRNA sequences. This rule comparison can help RNAi designers select the appropriate rules.

Although combined kernels improve prediction accuracy over single kernels, the improvements depend on which kernels are combined. Rules that performed well, such as Huesken's motifs and Amarzguioui's rule, already captured crucial properties in the sequences. Combining their numerical kernels with string kernels did not improve performance as drastically as combining less well performed rules. For example, a Gaussian kernel using Amarzguioui's rule yielded an average (over the four data sets) R of 0.497. An average R of 0.537 was obtained by combining it with RNK using the F-heuristic, and an average R of 0.518, when combined with RSK. These two improvements only increased the correlation coefficients by 8% and 4%, respectively. In contrast, Reynolds' rule, which did not perform so well alone, improved performance dramatically when combined with a string kernel. For instance, a Gaussian kernel on Reynolds' rule yielded an average R of 0.354. And combining it with RNA and RSK yielded average R of 0.529 and 0.526, increasing R by 47% and 46%, respectively, much higher than those gained by combining Amarzguioui's rule.

We believe the multikernel framework is applicable to other applications where data have heterogeneous components. One example is disease diagnosis, where microarray, clinical, and SNP data are available yet different. The kernel combination coefficients are easy to compute and may possibly increase the diagnostic accuracy.

Acknowledgement. This work was supported by NIH grant 1P20RR18754 from the Institutional Development Award (IDeA) Program of NCRR. Dr. Lane's work was also partially supported by NIMH grant 1R01MH076282-01 of the NSF/NIH Collaborative Research in Computational Neuroscience Program.

References

1. Hannon, G.J.: RNA interference. Nature 418, 244–251 (2002)
2. Check, E.: Hopes rise for RNA therapy as mouse study hits target. Nature 432, 136 (2004)
3. Brummelkamp, T.R., Bernards, R., Agami, R.: A system for stable expression of short interfering RNAs in mammalian cells. Science 296, 550–553 (2002)
4. Pei, Y., Tuschl, T.: On the art of identifying effective and specific siRNA. Nature Methods 3(9), 670–676 (2006)
5. Amarzguioui, M., Prydz, H.: An algorithm for selection of functional siRNA sequences. B.B.R.C. 316, 1050–1058 (2004)
6. Reynolds, A., Leake, D., Boese, Q., Scaringe, S., Marshall, W.S., Khovorova, A.: Rational siRNA design for RNA interference. Nature Biotechnology 22, 326–330 (2004)

7. Ui-Tei, K., Naito, Y., Takahashi, F., Haraguchi, T., Ohki-Hamazaki, H., Juni, A., Ueda, R., Saigo, K.: Guidelines for the selection of highly effective siRNA sequences for mammalian and chick RNA interference. Nucleic Acids Research 32, 936–948 (2004)
8. Jagla, B., Aulner, N., Kelly, P., Song, D., Volchuk, A., Zatorski, A., Shum, D., Mayer, T., Angelis, D.D., Ouerfelli, O., Rutishauser, U., Rothman, J.: Sequence characteristics of functional siRNAs. RNA 11, 864–872 (2005)
9. Huesken, D., Lange, J., Mickanin, C., Weiler, J., Asselbergs, F., Warner, J., Meloon, B., Engel, S., Rosenberg, A., Cohen, D., Labow, M., Reinhardt, M., Natt, F., Hall, J.: Design of a genome-wide siRNA library using an artificial neural network. Nature Biotechnology 23(8), 995–1001 (2005)
10. Ge, G., Wong, G., Luo, B.: Prediction of siRNA knockdown efficacy using artificial neural network models. Biochem Biophys. Res. Comm. 336, 723–728 (2005)
11. Sætrom, P., Snøve Jr., O.: A comparison of siRNA efficacy predictors. Biochemical and Biophysical Research Communications 321, 247–253 (2004)
12. Qiu, S., Lane, T., Buturovic, L.: A randomized string kernel and its applications to RNA interference. In: Proc. 22 AAAI Conference on Artificial Intelligence, Vancouver, BC, Canada, pp. 627–632. AAAI Press, Menlo Park (2007)
13. Teramoto, R., Aoki, M., Kimura, T., Kanaoka, M.: Prediction of siRNA functionality using generalized string kernel and support vector machine. FEBS Lett. 579, 2878–2882 (2005)
14. Jia, P., Shi, T., Cai, Y., Li, Y.: Demonstration of two novel methods for predicting functional siRNA efficiency. BMC Bioinformatics 7, 271 (2006)
15. Vert, J.P., Foveau, N., Lajaunie, C., Vandenbrouck, Y.: An accurate and interpretable model for siRNA efficacy prediction. MBC Bioinformatics 7, 520 (2006)
16. Vapnik, V.N.: Statistical Learning Theory. John Wiley and Sons, Chichester (1998)
17. Qiu, S., Lane, T.: The RNA string kernel for siRNA efficacy prediction. In: Proc. 7th IEEE Int'l Conf. on Bioinformatics and Bioengineering (BIBE 2007), Boston, MA, pp. 307–314 (October 2007)
18. Qiu, S., Adema, C., Lane, T.: A computational study of off-target effects of RNA interference. Nucleic Acids Research 33, 1834–1847 (2005)
19. Qiu, S., Lane, T.: RNA string kernels for RNAi off-target evaluation. Int. J. Bioinformatics Research and Applications (IJBRA) 2(2), 132–146 (2006)
20. Lanckriet, G.R.G., Cristianini, N., Bartlett, P., Ghaoui, L.E., Jordan, M.I.: Learning the kernel matrix with semidefinite programming. J. Machine Learning Research 5, 27–72 (2004)
21. Qiu, S., Lane, T.: Multiple kernel learning for support vector regression. Technical Report TR-CS-2005-42, Computer Science Department, The University of New Mexico, Albuquerque, NM, USA (2005)
22. Cristianini, N., Shawe-Taylor, J., Elissee, A., Kandola, J.: On kernel-target alignment. In: Dietterich, T., Becker, S., Ghahramani, Z. (eds.) Advances in Neural Information Processing Systems, vol. 14, MIT Press, Cambridge (2002)
23. Smola, A., Schölkopf, B.: A tutorial on support vector regression. Technical Report NC2-TR-1998-030, NeuroCOLT2 (1998)
24. Weston, J., Schölkopf, B., Eskin, E., Leslie, C., Noble, W.S.: A kernel approach for learning from almost orthogonal patterns. In: Elomaa, T., Mannila, H., Toivonen, H. (eds.) PKDD 2002. LNCS (LNAI), vol. 2431, Springer, Heidelberg (2002)
25. UCI: UCI machine learning data datasets, `http://www.ics.uci.edu/~mlearn/MLRepository.html`

Hierarchical Clustering Support Vector Machines for Classifying Type-2 Diabetes Patients

Wei Zhong[1,*], Rick Chow[1], Richard Stolz[3],
Jieyue He[4], and Marsha Dowell[2]

[1]Divison of Math and Computer Science, [2] School of Nursing
[3] School of Business Administration and Economics
University of South Carolina Upstate, Spartanburg, SC 29303, USA
wzhong@uscupstate.edu
[4] School of Computer Science and Engineering
Southeast University, Nanjing 210096, China

Abstract. Using a large national health database, we propose an enhanced SVM-based model called Hierarchical Clustering Support Vector Machine (HCSVM) that utilizes multiple levels of clusters to classify patients diagnosed with type-2 diabetes. Multiple HCSVMs are trained for clusters at different levels of the hierarchy. Some clusters at certain levels of the hierarchy capture more separable sample spaces than the others. As a result, HCSVMs at different levels may develop different classification capabilities. Since the locations of the superior SVMs are data dependent, the HCSVM model in this study takes advantage of an adaptive strategy to select the most suitable HCSVM for classifying the testing samples. This model solves the large data set problem inherent with the traditional single SVM model because the entire data set is partitioned into smaller and more homogenous clusters. Other approaches also use clustering and multiple SVM to solve the problem of large datasets. These approaches typical employed only one level of clusters. However, a single level of clusters may not provide an optimal partition of the sample space for SVM trainings. On the contrary, HCSVMs utilize multiple partitions available in a multilevel tree to capture a more separable sample space for SVM trainings. Compared with the traditional single SVM model and one-level multiple SVMs model, the HCSVM Model markedly improves the accuracy for classifying testing samples.

Keywords: Hierarchical Clustering, Support Vector Machines, Classification, Clustering Algorithm, Type-2 Diabetes.

1 Introduction

1.1 Motivations

The rapid rate of increase in people diagnosed with diabetes warrants immediate attention of policy makers and health care providers alike. According to the American

* Corresponding author.

I. Măndoiu, R. Sunderraman, and A. Zelikovsky (Eds.): ISBRA 2008, LNBI 4983, pp. 379–389, 2008.

Diabetes Association, an estimated 15 million Americans (more than 1 out of 20) have been diagnosed with diabetes. As the fifth leading cause of death, diabetes has its greatest effects on the elderly and certain racial/ethnic groups and is a major contributor to the escalating costs of health care that increased from about $8 billion in 1992 to $132 billion in 2002 [12].

Classification of patients diagnosed with type-2 diabetes based on the length of hospital stay is an important aspect of public health policy. Good understanding about how the length of stay is related to patient and hospital profiles will provide relevant clinical knowledge for clinicians and health policy experts to identify, evaluate, and subsequently predict the predominant variables that affect length of stay [7].

Current strategies to address problems in health care are based on small and localized data sets. Typically, only local benchmarks are used in these models [4], reducing their applicability to the larger and more general population. In contrast, this study utilizes The Healthcare Cost & Utilization Project (HCUP-3) databases which are the largest and most robust U.S. inpatient databases. Classification of patients diagnosed with type-2 diabetes based on the length of stay from national databases can provide important information for making effective healthcare policies at national, state, and local levels. If patients stay in a hospital more than nine days (the sample median), these patients are classified as "negative"; otherwise, these patients are classified as "positive".

Traditional approaches based on Support Vector Machines (SVMs) [14] are used to solve similar classification problems as in this study. SVMs are based on the idea of mapping data points to a high dimensional feature space where a separating hyper-plane can be found. SVMs search the optimal separating hyper-plane by solving a convex quadratic programming (QP) problem. The typical running time for a convex quadratic programming problem is $\Omega(n^3)$ for a training set with n samples. Convex quadratic programming problems are NP-complete in the worst case [15]. Therefore, a single SVM are not favorable for large datasets having complex distribution patterns [5]. For example, it took several weeks to train SVMs on our dataset containing thirty thousand records. SVMs also show difficulties in reaching convergence during the training process for datasets with complex distribution patterns.

1.2 Previous Work

Many algorithms and implementation techniques have been developed to enhance SVMs in order to increase their training performance on large data sets that have complex distribution patterns. The most well-known techniques include chunking [11], Osuna's decomposition method [8], and Sequential Minimal Optimization (SMO) [10]. The success of these methods depends on dividing the original Quadratic Programming (QP) problem into a series of smaller computational sub-problems. Although these algorithms accelerate the training process, they do not scale well with the size of the training data.

Another class of SVM enhancing algorithms tries to speed up the training process by providing SVMs with high quality training data. For example, data points such as the support vectors are more important to determine the optimal solution [1]. Random Selection [1][3], Bagging [13], and clustering analysis [2][6][17] are representatives of these algorithms. Although these algorithms are highly scalable for large datasets

with shorter training time, the accuracy of the trainings still depends greatly on the selection of training samples.

More recently, several one-level multiple SVM approaches are proposed to deal with trainings on large datasets [19]. One of approaches is called Clustering Support Vector Machines (CSVMs) in which the whole data set is partitioned into multiple clusters using a clustering algorithm. Then, individual CSVMs are trained for each of the clusters. Finally, during the testing phase, the clustering algorithm chooses one of the CSVMs to classify a testing sample. CSVMs use the capabilities of only one level of SVMs based on arbitrary partitions of data. Some SVMs trained on clusters obtained from poor partitions may develop poor classification capabilities. As a result, the classification capability of the CSVM model is not optimized.

1.3 New Approach

To further enhance the performance of one-level CSVMs, we propose a new computational model called Hierarchical Clustering Support Vector Machines (HCSVMs) that utilizes multiple levels of SVMs in a clustering tree. Using the hierarchical clustering technique, the whole dataset is first partitioned into multiple clusters at multiple levels of a tree-structure. This tree-structure represents multiple levels of partitions with widely different sizes, shapes, and densities. Each level of partitions can capture different data distribution patterns for a particular subspace of the data. In the second step, SVMs are built on clusters at each level so that they can perform classifications in different sample subspaces. Some partitions at some levels in the tree may provide more separable sample subspaces that enhance SVM trainings for certain subproblems. However, some partitions in the tree may not be as effective in capturing the right subspaces. One of the SVMs in the tree is most suitable for classifying testing samples of a given subspace. Such a SVM is found by adaptively searching through the tree. In this paper, we demonstrate the multi-level HCSVM approach has a superior classification performance as compared to the traditional single SVM model and the one-level CSVM model.

Our paper is organized as follows. In Section 2, background and details of two-class Support Vector Machines, Clustering Support Vector Machines, and Hierarchical Clustering Support Vector Machines are explained. In Section 3, the training data set and the testing data set are discussed. Experimental results and analysis are examined. Conclusions are presented in the last section.

2 Method

2.1 Support Vector Machines (SVMs)

The SVM is a promising computing model for solving classification and regression problems, based on convex Quadratic Programming [14]. Since SVMs are not favorable for a large dataset training with complex distribution patterns [5], One-level Clustering Support Vector Machines were proposed to enhance SVM training efficiency for large datasets [19].

2.2 One-Level Clustering Support Vector Machines (CSVMs)

In the one-level CSVM based approach, the training dataset is first divided into multiple clusters. A SVM is trained for each of the clusters to model the nonlinear relationships between the procedural and diagnostic profiles and the length of stay for Type-2 diabetes patients. Each SVM can focus on learning the distribution pattern of one homogenous cluster for effective knowledge discovery.

After the SVMs are trained, a new testing sample is assigned to a cluster based on the minimum *sample-cluster distance* defined below as the average distance between that sample and each sample in a given cluster. All samples in this study are encoded as binary vectors. The *sample-cluster distance* between a sample x and a given cluster is:

$$dist(C_i, x) = \frac{1}{n_i} \sum_{q \in C_i} dist(x, q) \tag{1}$$

where C_i is the cluster i, x is the given data sample, q is one of the samples in C_i, n_i is the number of samples in cluster C_i, and $dist(x, q)$ is the distance between sample x and sample q. Since all features of sample x and sample q are coded as binary numbers, $dist(x, q)$ is formulated as:

$$dist(x, q) = \frac{Match_{11}}{Match_{11} + Match_{01} + Match_{10}} \tag{2}$$

where $Match_{11}$ is the number of features where sample x is 1 and sample q is 1, $Match_{01}$ is the number of features where sample x is 0 and sample q is 1 and $match_{10}$ is the number of features where sample x is 1 and sample q is 0 [9]. Hence, the function for assigning a testing sample x to a selected cluster C_j is formulated as:

$$dist(C_j, x) = \min_{i=1,\ldots,n} dist(C_i, x) \tag{3}$$

The SVM classification function for the selected cluster C_j to classify the samples x is formulated as:

$$f_{svm\_j}(x) = \left(\sum_{i=1}^{sv} \alpha_i y_i K_{svm\_j}(x, x_i) + b\right) \tag{4}$$

where sv is the number of support vectors and $K_{svm\_j}(x, x_i)$ is the kernel function from *svm_j* trained for cluster C_j.

Although Clustering Support Vector Machines have produced some promising results [19], they focus on only one level of clusters produced by an arbitrary partitioning process in which the cluster sizes and cluster level are determined arbitrarily. As a result, some clusters may hinder SVM trainings because samples in these clusters could be non-separable. In the following section, we propose an enhanced SVM-based model called Hierarchical Clustering Support Vector Machine (HCSVM) that utilizes multiple levels of clusters to represent different abstractions of data. The advantage of multi-level clusters is that some clusters at the lower levels provide more suitable sample subspaces for SVM training than clusters at the upper levels and vice

versa. As a result the clusters in the lower levels and upper levels of the tree structure may produce SVMs with different classification capabilities. Consequently, the most suitable SVM is selected from the group of SVMs to generate the best classification response for a given sample as described in the following sections.

2.3 Hierarchical Clustering Support Vector Machines (HCSVMs)

In the first step of building the HCSVMs model, a large dataset is partitioned into multi-level clusters in a tree-structure using agglomerative hierarchical clustering techniques [9]. Second, a HCSVM is trained for each of the clusters. Finally, the HCSVMs at multiple levels work cooperatively to classify the samples.

2.3.1 Data Partitioning Using Hierarchical Clustering

In the first step of the agglomerative hierarchical clustering technique, each sample is considered as an individual cluster. At each step, the closest pair of clusters with the shortest distance defined by equation 5 is merged to form a bigger cluster. The process is repeated until the distance between two nearest clusters exceeds a specified threshold. The average distance between two clusters C_i and C_j, $dist(C_i, C_j)$, is defined as the average pair-wise distance of all pairs of samples from two clusters [9]:

$$dist(C_i, C_j) = \frac{1}{n_i n_j} \sum_{p \in C_i} \sum_{q \in C_j} dist(p, q) \tag{5}$$

where $dist(p, q)$ is the distance between samples p and q as defined in equation 2, n_i is the number of samples in cluster C_i, and n_j is the number of samples in cluster C_j. Such a distance measure is less susceptible to noise and outliers [9].

After the clustering process is complete, multiple clusters are located at different levels of the tree. Unlike the K-mean clustering algorithm, the hierarchical clustering algorithm does not assume a fixed number of clusters; instead, the hierarchical clustering algorithm generates appropriate numbers of clusters at different levels based on the underlying distribution patterns. Clusters at different levels are capable of capturing different levels of abstractions of the sample space.

2.3.2 HCSVM Training

SVMs are trained for clusters at different levels of the cluster tree. Some sample subspaces in the tree may enhance SVM trainings whereas some sample subspaces may worsen SVM trainings. As a result, SVMs at different levels may develop different classification capabilities because they are trained for different sample subspaces.

The time complexity for SVM training is $\Omega(n^3)$, where n is the number of samples in the training data set. In this approach, a SVM model is trained for each of the clusters at different levels of the tree-structure. During the training process, the parameters for a SVM model are optimized using grid search techniques [19]. Since the size of a typical cluster is much smaller than the size of the original data set and the trainings for different clusters can be performed in parallel, the overall training time is reduced substantially. For example, the original data set has about 5,000 samples and the largest cluster has about 500 samples.

2.3.3 Collaborative Classification Using HCSVM

After SVM trainings, individual SVMs are specialized to solve classification sub-problems in their own subspace. Given a set of test samples, an adaptive algorithm is used to select the most suitable SVMs in the tree to perform robust and effective classifications.

The classification value, $f_{svm\_j}(x)$, of a SVM, *svm_j*, is normalized using the z-score for fair comparison of classification values from different SVMs since classification values from different SVMs have different means and standard deviations. The *decision value* of *svm_j* for a sample *x* is defined as the z-score of *svm_j*'s classification value for a sample *x*:

$$Decision\_value_{svm\_j}(x) = \frac{(f_{svm\_j}(x) - mean_{svm\_j})}{\sigma_{svm\_j}} \tag{6}$$

where $mean_{svm\_j}$ is the mean classification values for *svm_j* in the training set and σ_{svm_j} is the standard deviation of classification values for *svm_j* in the training set [9].

The higher the magnitude of the *decision value* of a *SVM_j*, $|Decsion\_value_{svm\_j}(x)|$, will be the higher the SVM's confidence level for classifying a sample *x*. For example, if $Decison\_value_{svm\_j}(x)$ = -4.5 and $Decision\_value_{svm\_k}(x)$ = 3.2, then *svm_j* has a higher confidence level for its decision than *svm_k*. Hence, *svm_j* should be selected as the classifier to classify sample *x* as negative. The HCSVM model uses this confidence based selection strategy to select the best SVMs from the entire tree structure to produce the most confident classification results.

The HCSVM classification process may be described as a recursive, bottom-up process that operates on tree structures. Recall that the clusters are organized in a tree structure as shown in Figure 1. The classification process treats a subtree of clusters as a computing group with a root cluster, C_{root}, and its children clusters, C_i's. In turn, each of the children clusters is the root for its own subtree or computing group (Fig. 1).

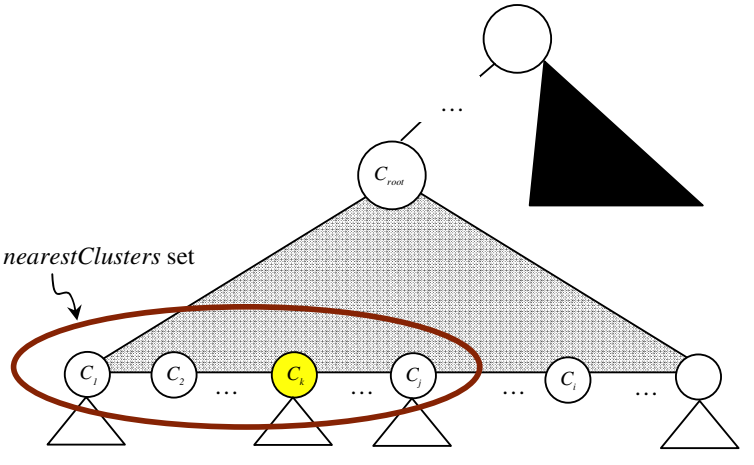


Fig. 1. Tree Structure for HCSVM Classification

When a cluster C_{root} that has no children receives a sample x to be classified, it simply reports to its parent cluster that its own decision value, $decision\_value_{svm\_root}$, as the decision value for its subtree. However, if C_{root} has some children clusters, C_{root} will first compare the distance between x and its children clusters and select a set of clusters that are nearest to x, denoted as the *nearestClusters* set in Figure 1. The *nearestClusters* set consists of clusters that are highly relevant for classifying the sample x. C_{root} then asks the children clusters to report the decision values of their subtrees. Suppose the subtree decision value of cluster C_k is the most confident decision value among all clusters in the *nearestClusters* set. The final decision value of C_{root}'s subtree is determined as either $decision\_value_{svm\_root}(x)$ or the subtree decision value of C_k, whichever is more confident. This process is recursive because the children clusters generate decision values for their own subtrees using the same steps. The whole process terminates when the topmost cluster generates the final decision value. The whole classification algorithm is shown in Figure 2.

The following notations are used in the pseudo codes as shown in Figure 2. Given a sample x, the SVM decision value for the root cluster is defined as $decision\_value_{svm\_root}(x)$. The term, $dist(C_i, x)$, is the distance between a sample x and cluster C_i as defined previously. The magnitude of the most confident SVM decision value for the sub_tree_k is defined as $|subtree\_Decision_k|$. The input parameters of the recursive function **tree_Decison()** is the root cluster, C_{root}, and a sample x whereas the output of this function is the most confident SVMs' decision value.

```
svm_Decision tree_Decision(TreeNode C_root , Sample x)
{
   if(C_root has no children)
     return Decision_value_svm_root (x);
   else
   {
      for each of the children cluster C_i of C_root
         compute dist(C_i , x);

      Let nearestClusters be the set, {C_j | dist(C_j , x) ≤ ε}, where ε is some threshold;

      for each of the cluster C_j ∈ nearestClusters
         subtree_Decision_j = tree_Decision(C_j , x);

      Select subtree_Decision_k s.t. |subtree_Decision_k| = max_j(|subtree_Decision_j|)

      if (|subtree_Decision_k| ≥ |decision_value_svm_root(x)|)
         return subtree_Decision_k ;
      else
         return decision_value_svm_root(x);
   }
}
```

Fig. 2. Recursive Function to Classify Sample Based on Cluster Tree Structure

3 Experimental Setup and Result Analysis

3.1 Comparisons of Three SVM Based Models

Experiments are set up to compare the performance of three SVM based models. The first model is the HCSVM model that uses multiple SVMs and multilevel clustering. The second model is CSVM that utilizes multiple SVMs with only one level of clusters. The third model is the single SVM model without any clustering. For the HCSVM trainings, hierarchical clustering is performed on the training dataset to create a tree of clusters. The HCSVM trainings involve SVM trainings for clusters at different levels throughout the entire tree. A total of 36 HCSVMs are trained in this experiment. In the control experiment, the CSVM trainings use only a single level of 11 clusters; hence, only 11 SVMs are trained.

3.2 Training Set and Independent Test Set

This study uses the Healthcare Cost & Utilization Project (HCUP-3) database provided by the Agency for Healthcare Research and Quality (US department of Health and Human Service). HCUP-3 is the largest and most robust U.S. inpatient database with more than 600 clinical and non-clinical variables for each hospital record. The training set consists of 5,000 samples for diabetes patients with at least two procedural codes and the independent test set has 1,500 samples for diabetes patients with at least two procedural codes.

3.3 Selected Feature for SVM Training and Testing

Features of each sample are grouped into four major categories including patient profiles, hospital profiles, diagnostic profiles and procedure profiles. A patient profile consists of demographic and insurance information. A hospital profile includes hospital type, teaching status, and location. A diagnostic profile consists of a set of diagnostic codes for patients. The procedure profile consists of a set of treatment codes for patients. Our initial experiment shows that the combination of diagnostic and procedural profiles gives the most reliable HCSVM prediction results. As a result, only diagnostic and procedural codes of each sample are used for SVM trainings.

3.4 Results and Analysis

Figure 3 compares average accuracy of the HCSVM model, the CSVM model and the single SVM model. Average accuracy for the HCSVM has improved from 63% to 73% compared to the CSVMs model.

Based on the difficulty of classifying the data, the independent testing set is separated into three groups. The “bad data group” includes data that CSVM classifies with testing accuracy below 60%. The “average data group” includes data that CSVM classifies with testing accuracy between 60% and 70%. The “good data group” includes data that CSVM classifies with testing accuracy over 70%. In other words, the bad data group is the portion of data poorly classified by CSVM in the testing set. The good data group is the portion of data accurately classified by CSVM in the testing set.

Figure 4 compares the performance of the HCSVM model, the CSVM model and the single SVM model for the bad data group, average data group and good data group. For the bad data group, accuracy improves from 50% to 63% when the HCSVM

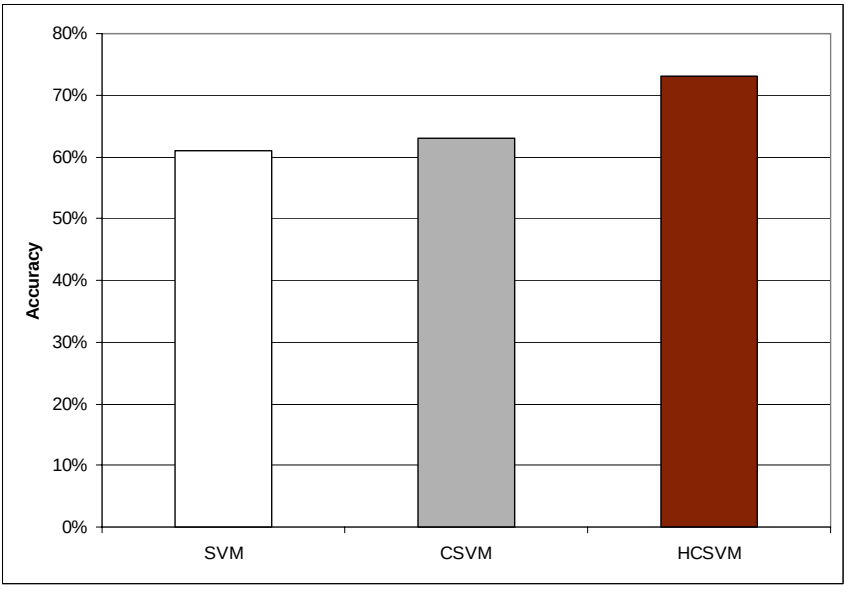


Fig. 3. Average Accuracy of SVM, CSVMs and HCSVM for Entire Testing Set

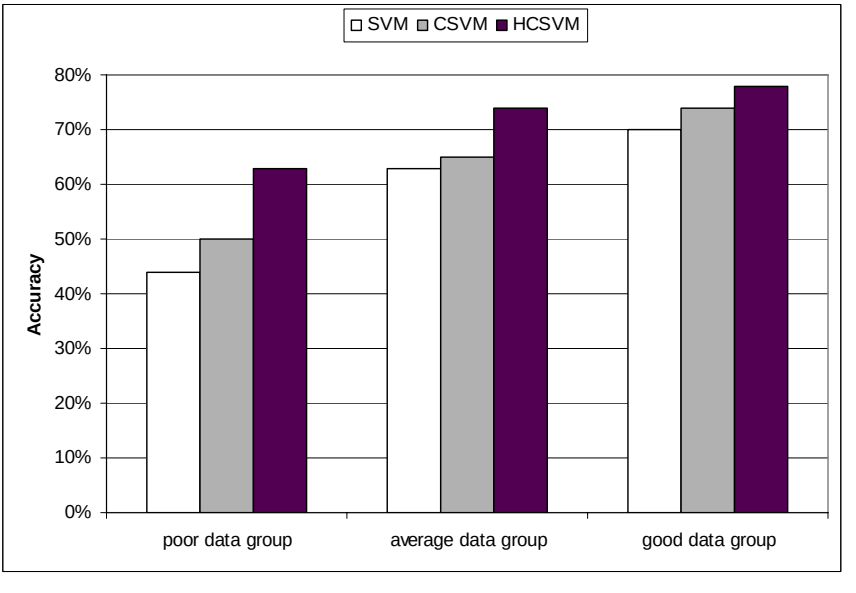


Fig. 4. Accuracy of SVM, CSVMs and HCSVM for Different Data Groups

model is used. For the good data group, accuracy increases by 4%. The performances of both CSVM and HCSVM for the three data groups are better than the performance of the single SVM model. This indicates that the strategy of building multiple SVMs in the training set worked. In the tree structure, all HCSVMs at different levels cooperate to make intelligent and robust classification decisions. Our algorithm chooses the most confident HCSVM in the tree structure for the final classification decision. This strategy of fully utilizing the capability of all SVMs at every level of the tree structure is demonstrated to be highly effective for classifying samples than both the single SVM model and the one-level CSVMs model.

4 Conclusion

In this study, a new computational model called HCSVM converts a complex classification problem into a series of computational subproblems for large datasets. Each HCSVM is customized to learn the different nonlinear relationships between diagnostic and procedural profiles and the length of stay for patients in each cluster. The HCSVM model uses a confidence-based selection strategy to select the most suitable SVM to produce the most accurate response.

With the CSVM model, only one level of SVMs is used to classify samples. Clusters produced by poor partitions may not help SVMs find a good separating hyperplane in the CSVM model. In contrast, the HCSVM model uses an adaptive strategy to select the most confident SVMs that uses more optimal partitions to find a better hyperplane for effective classifications. Experimental results indicate that the performance of the multi-level HCSVM model is superior to the one-level CSVM model for the classification problem.

Acknowledgments. This research was supported in part by Student Research Assistant Program and Research Incentive Award from University of South Carolina Upstate.

References

1. Agarwal, D.K.: Shrinkage estimator generalizations of proximal support vector machines. In: Proc. of the 8th ACM SIGKDD international conference of knowledge Discovery and data mining, Edmonton, Canada (2002)
2. Award, M., Khan, L., Bastani, F., Yen, I.: An Effective Support Vector Machines(SVMs) Performance Using Hierarchical Clustering. In: Proc. of the 16th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2004) (2004)
3. Balcazar, J.L., Dai, Y., Watanabe, O.: Provably Fast Training Algorithms for Support Vector Machines. In: Proc. of the 1st IEEE International Conference on Data mining, pp. 43–50. IEEE Computer Society, Los Alamitos (2001)
4. Breault, J.L., Goodall, C.R., Fos, P.J.: Data Mining a Diabetic Data Warehouse. Artificial Intelligence in Medicine 26, 37–54 (2002)
5. Chang, C.C., Lin, C.J.: Training nu-support vector classifiers: Theory and algorithms. Neural Computations 13, 2119–2147 (2001)

6. Daniael, B., Cao, D.: Training Support Vector Machines Using Adaptive Clustering. In: Proc. of SIAM International Conference on Data Mining, Lake Buena Vista, FL, USA (2004)
7. Dowell, M.A., Rozell, B., Roth, D., Delugach, H., Chaloux, P., Dowell, J.: Economic and Clinical Disparities in Hospitalized Patients with Type-2 Diabetes. Journal of Nursing Scholarship 36, 66–72 (2004)
8. Osuna, E., Freund, R., Girosi, F.: An improved training algorithm for support vector machines. In: Proc. Of IEEE Workshop on Neural Networks for Signal Processing, pp. 276–285 (1997)
9. Jain, A.K., Murty, M.N., Flynn, P.J.: Data clustering: A review. ACM Computing Surveys 31, 264–323 (1999)
10. Platt, J.: Fast training of support vector machines using sequential minimal optimization. In: advances in Kerenel Methods-Support Vector Learning, pp. 185–208 (1999)
11. Scholkopf, B., Burges, C., Smola, A. (eds.): Advances in Kernel Methods-Support Vec-tor Learning. MIT Press, Cambridge, MA (1999)
12. US Department of Health and Human Services, Centers for Disease Control and Prevention: Prevalence of diabetes and impaired fasting glucose in adults-United States 1999–2000, Morbidity and Mortality Weekly Report 52, 833–835 (2003)
13. Valentini, G., Dietterich, T.G.: Low Bias Bagged Support vector Machines. In: Proc. of the 20th International Conference on Machine Learning ICML 2003, Washington D.C. USA, pp. 752–759 (2003)
14. Vapnik, V.: Statistical Learning Theory. John Wiley & Sons, Inc., New York (1998)
15. Vavasis, S.A.: Nonlinear Optimization: Complexity Issues. Oxford Science, New York (1991)
16. Yao, Y.Y.: Perspectives of Granular Computing. In: IEEE Conference on Granular Computing (to appear, 2005)
17. Yu, H., Yang, J., Han, J.: Classifying Large Data sets Using SVMs with Hierarchical Clusters. In: Proc. Of the 9th ACM SIGKDD 2003 (2003)
18. Zagrovic, B., Pande, V.S.: How does averaging affect protein structure comparison on the ensemble level? Biophysical Journal 87, 2240–2246 (2004)
19. Zhong, W., He, J., Harrison, R., Tai, P.C., Pan, Y.: Clustering Support Vector Machines for Protein Local Structure Prediction. Expert Systems With Applications 32, 518–526 (2007)

Computational Mutagenesis of *E. coli Lac* Repressor: Insight into Structure-Function Relationships and Accurate Prediction of Mutant Activity

Majid Masso, Kahkeshan Hijazi, Nida Parvez, and Iosif I. Vaisman

Laboratory for Structural Bioinformatics, George Mason University, 10900 University Blvd. MS 5B3, Manassas, VA 20110, USA
{mmasso,ivaisman}@gmu.edu,kahk2001@yahoo.com,nidaparvez@hotmail.com

Abstract. A computational mutagenesis methodology that utilizes a four-body, knowledge-based, statistical contact potential is applied toward quantifying relative changes (*residual scores*) to sequence-structure compatibility in *E. coli lac* repressor due to single amino acid residue substitutions. We show that these residual scores correlate well with experimentally measured relative changes in protein activity caused by the mutations. The approach also yields a measure of environmental perturbation at every residue position in the protein caused by the mutation (*residual profile*). Supervised learning with a decision tree algorithm, utilizing the residual profiles of over 4000 experimentally evaluated mutants for training, classifies the mutants based on activity with nearly 79% accuracy while achieving 0.80 area under the receiver operating characteristic curve. A trained decision tree model is subsequently used to infer the levels of activity for all remaining unexplored *lac* repressor mutants.

Keywords: lac repressor Delaunay tessellation, statistical potential, computational mutagenesis, supervised learning.

1 Introduction

The *lac* repressor of *Escherichia coli* is a well-studied DNA-binding protein, and the results of laborious biochemical experiments have been summarized thoroughly with several review articles in the literature [1-5]. While *lac* repressor possesses the helix-turn-helix motif common to bacterial repressors, it diverges from many of its counterparts in that it is functional as a homotetramer rather than a dimer [6]. Full blockage of transcription initiation requires binding of the nearly perfect palindromic operator sequence O1, centered at position +11 in the *lac* operon, accompanied by binding of operators O2 and O3, located within 401 and 92 base-pairs from O1 [7]. The DNA-binding domain of the *lac* repressor, known as the headpiece, consists of amino acids 1-59 (Fig. 1A). The core, which covers residues 61-329 and includes structurally similar N- and C-terminal subdomains, contains sites for inducer binding [8, 9] as well as dimer formation [10]. Amino acids 61-160 and 293-320 form the N-terminal core domain; the remaining residues constitute the C-terminal core domain. Lastly, amino acids 330-360 contain a leucine minizipper required for tetramerization of the

I. Măndoiu, R. Sunderraman, and A. Zelikovsky (Eds.): ISBRA 2008, LNBI 4983, pp. 390–401, 2008.

dimers through formation of a four-helical bundle [11, 12]. In the presence of inducers such as allolactose or isopropyl-β-D-thiogalactoside (IPTG), *lac* repressor undergoes a conformational change upon inducer binding and no longer binds operator DNA, allowing *lac* gene transcription.

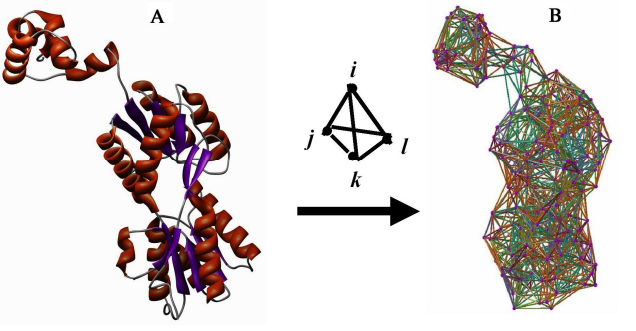


Fig. 1. (A) Ribbon diagram of a single chain of the *lac* repressor homotetramer. (B) Delaunay tessellation of the same monomer of *lac* repressor, subject to a 12 angstrom edge-length filter. PDB accession file: 1efa, chain B.

A wealth of structural and functional information about *lac* repressor has been discovered through the analysis of experimental data obtained from large-scale mutagenesis studies [13-15]. In particular, each wild-type (wt) amino acid located at positions 2-329 was individually replaced with 13 of the 20 amino acids naturally occurring in proteins. The physicochemical properties of each of the seven amino acids excluded as replacements from the experimental mutagenesis was accounted for by the existence of a similar residue from among the 13 substitutions. Although this approach led to 328 positions × 13 substitutions/position = 4264 substitutions, the wt amino acids at 223 positions in *lac* repressor were identical to one of the 13 residues chosen as replacements; hence, a total of 4041 non-degenerate single site mutants were experimentally synthesized for the studies. The phenotypic effects associated with each mutant were measured, and all 328 *lac* repressor residue positions were clustered into 15 groups based on their structural locations, functional roles, and level of tolerance to mutations [13, 14].

Here we characterize the single site mutants of *lac* repressor based on an application of a four-body statistical potential derived by means of Delaunay tessellation of protein structures [16, 17]. Using the method, a scalar *residual score* that quantifies the relative change in overall sequence-structure compatibility from wt was calculated for every such mutant. Focusing exclusively on the 4041 mutants of *lac* repressor for which there exists phenotypic data describing the effects of the residue replacements on activity, we illustrate how the residual scores of these mutants can be used to elucidate the structure-function relationship inherent in the protein. This approach has been successfully applied to other large systems of single site protein mutants, including HIV-1 protease (536 mutants) and reverse transcriptase (366 mutants), as well as bacteriophage T4 lysozyme (2015 mutants) [18]. Unlike the prior studies, which exclusively targeted enzymes, here we investigate a DNA-binding protein for the first time.

The computational mutagenesis that we developed also generated a vector *residual profile* representation for each mutant, where the vector components quantify the environmental perturbations from wt that occur at every residue position due to the mutation. Recent reports have described training sets of mutant protein feature vectors, produced for the purposes of supervised classification and inference [19-21]. The vector components in those studies represent information readily available from the sequence (e.g., physicochemical classes of wt and replacement residues, hydrophobicity difference, and mutated position conservation score), and information predicted from protein structure (e.g., secondary structure, buried charge, and solvent accessibility). As will be detailed in this manuscript, our residual profiles implicitly incorporate both sequence and structure information about each mutant. Analysis of the residual profiles for the 4041 experimental mutants of *lac* repressor via decision tree supervised learning suggests that they encode signals capable of being used to reliably distinguish between mutants belonging to distinct activity classes. As such, a model trained with these mutants was used to infer activity classes for the remaining uncharacterized *lac* repressor mutants based their respective residual profiles.

2 Materials and Methods

2.1 Experimental Data

The 4041 *lac* repressor mutants described in the literature were generated via suppression of amber mutations that were independently introduced into the *lacI* gene at each of 328 sites, corresponding to residue positions 2-329 in the *lac* repressor protein [14]. The level of activity of each mutant protein was measured by its ability to repress the synthesis of β-galactosidase, one of the gene products encoded by the *lac* operon. The investigators used four activity classes as a way to categorize the phenotypes: full activity (greater than 200 fold repression of β-galactosidase, i.e., no significant alteration), moderate activity (20-200 fold repression), low activity (4-20 fold repression), and inactive (less than 4 fold repression) [13-15]. These categories were chosen arbitrarily as a rough guide, and the same team of researchers suggested combining the moderate and low activity classes into a single intermediate class [15]. We utilized for our studies a preformatted tabulation of the *lac* repressor mutants and their corresponding phenotypes, available as one of the training sets for the SIFT algorithm [22].

2.2 Delaunay Tessellation and the Four-Body Statistical Potential

A non-homologous training set of over 1400 high-resolution crystallographic protein structures with low primary sequence identity was selected from the Protein Data Bank (PDB) [23]. Each structure was represented as a discrete set of points in 3-dimensional (3D) space, corresponding to the C_α atomic coordinates of each of the constituent amino acid residues. Delaunay tessellation was performed on each protein structure, whereby these points were utilized as vertices to generate an aggregate of non-overlapping, space-filling, irregular tetrahedral simplices (Fig. 1B) [16, 17]. The Quickhull algorithm [24] was used to tessellate each protein, and an in-house suite of Java and Perl programs were developed for data processing and analyses.

Each simplex in a protein structure tessellation objectively defines a quadruplet of nearest-neighbor residues. For added assurance of biochemically feasible quadruplet interactions, we only considered simplices in protein tessellations for which the lengths of all six edges were less than 12 angstroms. Assuming order independence, there are 8855 distinct quadruplets that can be formed from the 20 amino acids naturally occurring in proteins [16, 17]. For each quadruplet, we determined the observed proportion of simplices among all the protein tessellations whose vertices represented the four amino acids. We also computed a rate expected by chance for each quadruplet based on a multinomial reference distribution that utilized the frequency of each amino acid among the training set proteins. Modeled after the inverse Boltzmann law, an empirical potential of quadruplet interaction (log-likelihood score) was calculated as a logarithm of the ratio of observed to expected values. The four-body statistical potential is defined as the collection of 8855 quadruplet types along with each of their respective log-likelihood scores [16, 17].

Using the four-body statistical potential, a log-likelihood score was assigned to each simplex in the tessellation of *lac* repressor subject to a 12 angstrom edge-length filter. We performed the tessellation on a single chain of the *lac* repressor tetramer obtained from the PDB accession file 1efa, chain B, which includes atomic coordinates for amino acids 2-331 [25]. The *topological score* of *lac* repressor, defined by adding up the log-likelihood scores of all simplices in the tessellated structure, represents an overall measure of sequence-structure compatibility. A *residue environment score* was also calculated for each amino acid position by locally adding up only log-likelihood scores of simplices that utilize the corresponding C_α coordinate as a vertex. A vector of residue environment scores, ordered by primary sequence position, is referred to as a *3D-1D potential profile* (Fig. 2A) [18, 26].

2.3 Computational Mutagenesis

A topological score was also obtained for each single site mutant of *lac* repressor, by utilizing the tessellation of the wt protein structure as a template, substituting the amino acid identity at a point representing the position of interest, and recalculating. Such an approach results in changes to the log-likelihood scores of all simplices that use the point as a vertex. The *residual score* of a *lac* repressor mutant is defined as the difference in topological scores between the mutant and wt protein, and provides a measure of the relative change in sequence-structure compatibility caused by the amino acid replacement [18]. A *comprehensive mutational profile* (CMP) is defined by calculating, at each position in the protein, the mean of the residual scores associated with all possible amino acid replacements (Fig. 2B) [18, 26]. Each CMP profile component is referred to as the *CMP score* of the given position.

Replacing the amino acid identity at a vertex in the wt protein tessellation leads to altered residue environment scores at the mutated position as well as at all positions that participate with it in nearest-neighbor simplices [18, 26]. The *residual profile* of a *lac* repressor mutant is defined as the difference in 3D-1D potential profiles between the mutant and wt protein, and the value of each residual profile component is referred to as an *environmental change (EC) score*. Residual profiles contain implicit structural information, since the only non-zero EC scores occur at components corresponding to the mutated position and all its nearest neighbors. In particular, the EC

score at the component corresponding to the mutated position is precisely the residual score of the mutant. Additionally, residual profiles of mutants defined by alternative amino acid substitutions at the same position have identical arrangements of zero and non-zero components. However, the EC scores differ at the non-zero components of these residual profiles, implicitly reflecting the type of substitution. Hence structure and sequence information is encoded in mutant residual profiles.

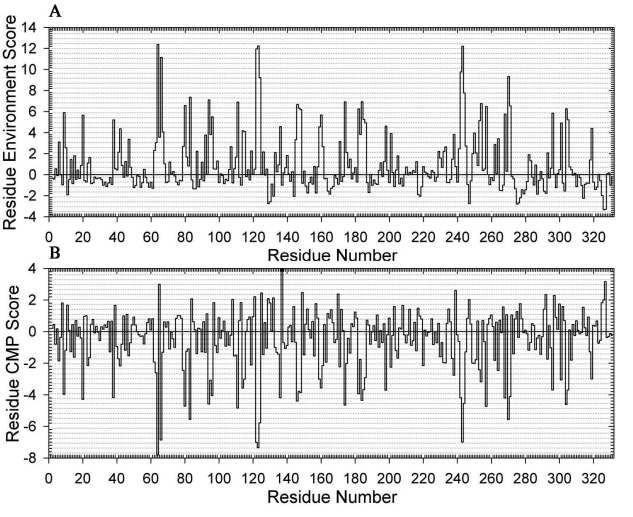


Fig. 2. *Lac* repressor (A) 3D-1D potential profile and (B) CMP profile

2.4 Supervised Learning for Class Discrimination and Prediction

The supervised classification scheme that we employed for this study is an implementation of Ross Quinlan's C4.5 decision tree algorithm [27], available as part of the Weka (Waikato environment for knowledge analysis) suite of machine learning tools [28]. Each of the experimental mutants in the training set was represented by a vector whose components consisted of the residual profile, the activity class to which the mutant belonged, and three additional components identifying the mutant (wt residue, position number, and replacement residue). Algorithm performance on the training set was evaluated by using stratified tenfold cross-validation (10 CV).

Given a generic two-class training set consisting of "positive" (P) and "negative" (N) examples, Q = accuracy = (TP + TN) / (TP + FN + FP + TN) provides a simple measure of 10 CV performance which is meaningful so long as the class distribution is not highly skewed. Here, TP and TN represent the number of correct positive and negative predictions, respectively, and FP and FN are misclassifications. The balanced error rate (BER), calculated as BER = 0.5 × [FN / (FN + TP) + FP / (FP + TN)], Matthew's correlation coefficient (MCC), given by

$$\mathrm{MCC} = \frac{\mathrm{TP} \times \mathrm{TN} - \mathrm{FP} \times \mathrm{FN}}{\sqrt{(\mathrm{TP}+\mathrm{FN})(\mathrm{TP}+\mathrm{FP})(\mathrm{TN}+\mathrm{FN})(\mathrm{TN}+\mathrm{FP})}},$$

and area (AUC) under the receiver operating characteristic (ROC) curve provide alternative measures that are especially useful for highly unbalanced classes. The ROC curve is a plot of the true positive rate (*sensitivity*) versus the false positive rate (1 - *specificity*), where sensitivity = TP / (TP + FN) and specificity = TN / (TN + FP), and the AUC is equivalent to the non-parametric Wilcoxon-Mann-Whitney test of ranks [29]. An AUC value near 0.5 suggests that the trained model will perform no better than random guessing, while a value of 1.0 is indicative of a perfect classifier. We utilize a conservative estimate for the standard error (SE) of the AUC that was derived by Hanley and McNeil [30]. For a training set consisting of examples that belong to more than two classes, a reference class is selected, and all examples not in this class are relabeled as belonging to the non-reference class. A two-class ROC analysis is then performed as described above, and the procedure is repeated so that each class serves as a reference. An overall AUC measure is calculated as a weighted average of the AUC values associated with each of the reference class ROC curves, where the AUC weights correspond to the proportion of examples that belong to the respective reference classes [31].

3 Results and Discussion

3.1 Structure-Function Correlation

Utilizing the 330 amino acids (positions 2-331) for which atomic coordinate data exists in the *lac* repressor PDB structure file 1efaB, theoretically there are 330 positions × 19 substitutions/position = 6270 possible single site mutants that can be generated. Although we computed the residual scores and residual profiles for all of them, here we consider only the 4041 experimental mutants for which the phenotypic effects of the substitutions are known. The distribution of the mutants among the four original activity classes is 2267 fully active, 253 moderate activity, 355 low activity, and 1166 inactive. As described earlier, we followed researcher suggestions by combining moderate and low activity groups into a single intermediate activity class, and we computed a mean residual score for the mutants in each class (Fig. 3, "All" category). A clear trend emerges, whereby increasingly detrimental effects on sequence-structure compatibility due to mutation are associated with higher levels of functional impairment. Furthermore, *t*-tests revealed statistically significant differences between mean residual scores for each pair of classes (fully active – intermediate, $p = 4.64\times10^{-7}$; intermediate – inactive, $p = 6.57\times10^{-10}$; fully active – inactive, $p = 1.95\times10^{-36}$). Within each class, mutants were also clustered based on whether they represented conservative (C) or non-conservative (NC) substitutions of the wt residue, and we computed mean residual scores for each of these subgroups [32]. Note that the overall trend is driven by NC substitutions, since C substitutions minimally impact sequence-structure compatibility regardless of phenotype.

Recently published computational studies investigating this dataset of experimental *lac* repressor mutants have focused specifically on whether an amino acid replacement has no effect (fully active class) or any detrimental effect (intermediate and inactive classes combined) on protein function [19-22, 33]. Similar trends are clearly maintained for this two-class system, and the difference in mean residual scores

between the unaffected and affected activity classes is statistically significant ($p = 8.67\times10^{-37}$). As mentioned earlier, such structure-function correlations based on mutant activity were observed for other proteins, and we hypothesized that the trend applied to all proteins [18]. These *lac* repressor findings bolster our prior claim.

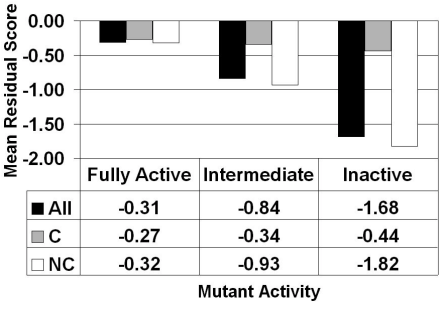


Fig. 3. *Lac* repressor structure-function correlation (see text for C/NC subsets)

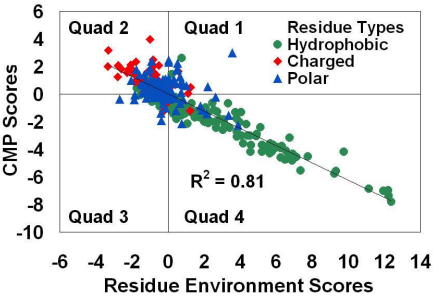


Fig. 4. *Lac* repressor CMP-potential profile correlation

3.2 Classification of Residue Positions

A strong inverse correlation ($R^2 = 0.81$) exists between the CMP profile of *lac* repressor, obtained by averaging the residual scores of all amino acid replacements at each position, and the 3D-1D potential profile of the protein, which provides an environment score for each position (Figs. 2 and 4). By averaging residual scores of non-conservative and conservative substitutions separately at each position, modified NC-CMP and C-CMP profiles showed that the correlation is driven primarily by the NC substitutions ($R^2 = 0.81$), with no contribution from the C substitutions ($R^2 = 0.09$). Similar observations have been made for HIV-1 protease [18, 26] and have been easily generated *in silico* for a number of other proteins (unpublished).

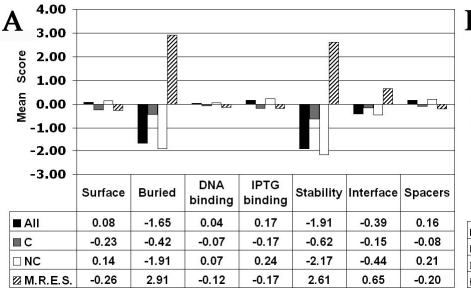
In particular for proteins with annotated residues, a majority of the buried hydrophobic amino acids tend to be located in the fourth quadrant (Q4) of plots analogous to Fig. 4, while many of the residues that make direct contacts with ligand (DNA-binding proteins) or are catalytic (enzymes) cluster in Q2. Additionally, other residues that are important for stability or that play functional roles (e.g. dimer interface or non-catalytic active site residues) are located between those extremes in the plot: stability residues tend to be found closer to those that are buried and hydrophobic, while functional residues are generally located near those that are ligand binding or catalytic. Catalytic residues generally tend to exhibit more extreme behavior (further away from the origin and toward upper left corner of Q2) than DNA-binding residues. Finally, remaining surface residues without these critical roles (mostly polar) tend to cluster near the origin. Q1 and Q3 typically contain relatively fewer positions than Q2 and Q4, and they are generally located closer to the origin.

Based on the extensive experimental work on *lac* repressor, all 328 residues comprising positions 2-329 were annotated and assigned to one of 15 groups according to structural locations and functional considerations [13]. It sufficed for us to work with a reduced set of seven groups, formed by pooling groups that shared similar properties.

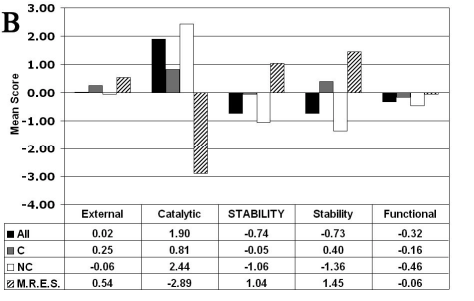
Table 1 provides a breakdown of the distribution of residue positions by group and quadrant location. Application of a chi-square test on this table led us to reject the null hypothesis that no association exists between the structural / functional groups and the quadrant locations ($p < 0.0001$). We also characterized each group based on both the mean of the residue environment scores (M.R.E.S) of the positions in the group, as well as the mean of the mutant residual scores (All, C, NC) for all 19 residue replacements at all positions in the group combined (Fig. 5A). It is clear from Fig. 5A that our computational characterization of these groups effectively discriminates DNA- and IPTG-binding residues from buried and stability positions. Another noteworthy observation is the fact that we can clearly distinguish between dimer interface residue positions and other surface residues that are not as structurally or functionally important.

Table 1. Distribution of *lac* repressor residue positions

Graph Quadrants	Residue Groups							
	Surface	Buried	DNA binding	IPTG binding	Stability	Interface	Spacers	Total
Q1	8	10	0	2	1	6	4	31
Q2	49	12	9	9	8	15	20	122
Q3	13	5	4	2	2	5	6	37
Q4	31	46	5	4	25	17	10	138
Total	101	73	18	17	36	43	40	328



	Surface	Buried	DNA binding	IPTG binding	Stability	Interface	Spacers
All	0.08	-1.65	0.04	0.17	-1.91	-0.39	0.16
C	-0.23	-0.42	-0.07	-0.17	-0.62	-0.15	-0.08
NC	0.14	-1.91	0.07	0.24	-2.17	-0.44	0.21
M.R.E.S.	-0.26	2.91	-0.12	-0.17	2.61	0.65	-0.20



	External	Catalytic	STABILITY	Stability	Functional
All	0.02	1.90	-0.74	-0.73	-0.32
C	0.25	0.81	-0.05	0.40	-0.16
NC	-0.06	2.44	-1.06	-1.36	-0.46
M.R.E.S.	0.54	-2.89	1.04	1.45	-0.06

Fig. 5. Characterization of (A) *lac* repressor and (B) HIV-1 reverse transcriptase (RT) residue positions based on structure and function

Motivation for such residue characterizations is due to earlier work with the enzyme HIV-1 reverse transcriptase (RT). Published mutagenesis experiments examined the effects of 366 single amino acid replacements among residues 95-203 (the fingers and palm subdomains) in the 66 kDa subunit of the RT heterodimer [34]. Based on the observations, a majority of the residues were annotated as being members of one of the following groups: catalytic, functional, STABILITY (strict), stability (liberal),

or external [34]. We implemented a computational mutagenesis procedure similar to that described above for *lac* repressor, which led to the graph for RT in Fig. 5B. Clear similarities exist between the RT and *lac* repressor graphs with respect to the functional/interface, stability, and external/surface categories; also noticeable are the much less extreme values for the DNA- and IPTG-binding residue groups in *lac* repressor compared to those for the catalytic residue group in RT.

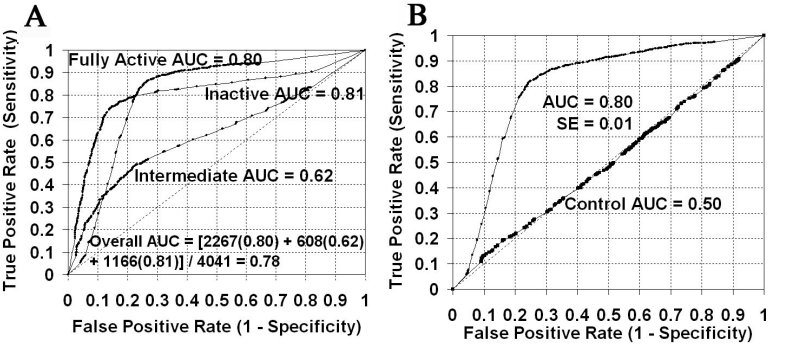


Fig. 6. ROC curves obtained using (A) three activity classes and (B) two classes

3.3 Inferential Models of Mutant Activity

Beginning with a training set consisting of the modified residual profiles (wt and new residues as well as mutated position number included, see Materials and Methods) for the 4041 experimental single site mutants, each belonging to one of three activity classes (fully active, intermediate, inactive), application of decision tree learning in conjunction with a 10 CV testing procedure led to Q = 73.0% and overall AUC = 0.78 (Fig. 6A). By relabeling the class of each mutant as either unaffected (fully active) or affected (intermediate and inactive) and applying 10 CV, we obtained Q = 78.7% and AUC = 0.80 ± 0.01 for this two-class system (Fig. 6B). Although the mutants are close to being equally distributed between the classes (2267 unaffected and 1774 affected), for completeness we also calculated BER = 0.22 and MCC = 0.57. A random shuffling of the activity class labels among the mutants in the two-class system prior to implementing 10 CV yields AUC = 0.50 and suggests that a decision tree model trained with this "shuffled classes" control is not expected to perform any better than random guessing (Fig. 6B). Additionally, with Q = 51.1%, BER = 0.51, and MCC = – 0.01, this random control highlights the strength of the embedded signals in the residual profiles forming the original training set.

Continuing with the two-class mutant system, our next aim was to measure the influence of the training set size on model accuracy. We began by applying decision tree learning and 10 CV to each of 10 stratified random samples of 100 training set mutants, where each subset was selected from among all 4041 experimental *lac* repressor mutants, and a mean accuracy and standard deviation (std. dev.) was calculated. Subsequent iterations involved incrementing by 100 mutants the size of the sampled training sets. Given the observed plateau in the learning curve generated

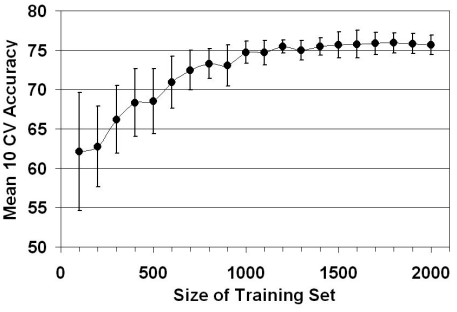


Fig. 7. Learning curve for the *lac* repressor training set with two class labels. Error bars represent ± 1 std. dev. from the mean.

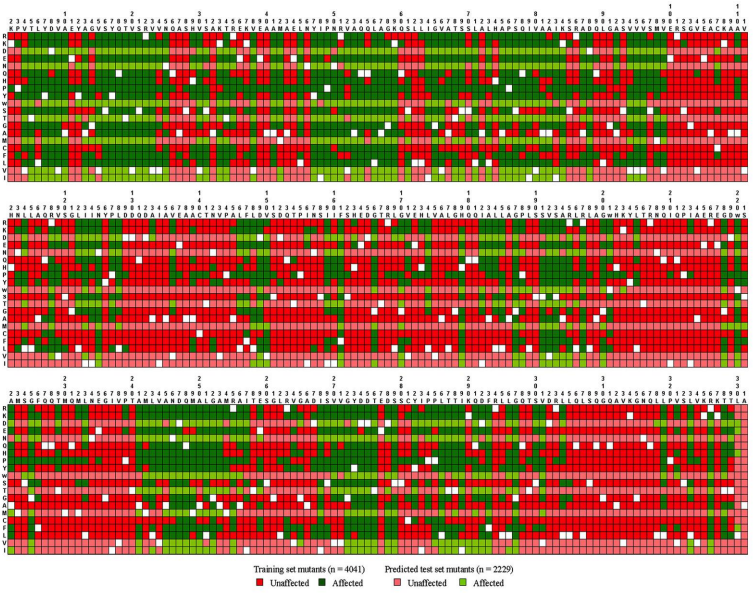


Fig. 8. *Lac* repressor mutational array. Columns – amino acid positions 2-331, Rows – twenty amino acid substitutions; Red – unaffected, green – affected, white – substitution identical to wt; darker colors – experimental, lighter colors – predicted.

from the data (Fig. 7), it was not necessary to increment the training set size beyond 2000 mutants. In fact, as suggested by Fig. 7, a decision tree model achieving optimal accuracy can be learned from approximately 1200 mutants.

To illustrate an important practical application, we employed a decision tree model learned from the entire training set of 4041 mutants in order to predict the unaffected/affected class memberships of all remaining uncharacterized single site *lac* repressor mutants. In particular, since we already calculated the residual profiles for all 6270 mutants, generated by the 19 single amino acid replacements at residue positions 2-331, there remained 2229 uncharacterized mutants whose residual profiles

were used to form a separate test set. Based on the signals encoded in their residual profiles, the decision tree model generated a class prediction for every test set mutant. We pooled all experimental and predicted *lac* repressor mutants at positions 2-331 into the array shown in Fig. 8, which summarizes overall mutational patterns in the protein. Columns represent residue positions in wt *lac* repressor; and rows represent the 20 possible amino acid replacements, arranged from top to bottom in order of increasing hydrophobicity [35]. Notably, at positions already determined to be either highly tolerant or intolerant of residue substitutions, our predictions for the uncharacterized residue replacements are well in line with the experimental findings.

Finally, 30% of the *lac* repressor mutants (slightly over 1200) were randomly selected to train a decision tree model, which is a minimally optimal training set as suggested by the learning curve (Fig. 7). All the remaining mutants formed a separate test set, and the model was used to predict their class memberships. Out of 1586 unaffected mutants in the test set, 1316 were correctly predicted; similarly, out of 1243 affected mutants in the test set, 873 were correctly predicted. Based on these results, we obtained Q = 77.4%, BER = 0.23, MCC = 0.54, and AUC = 0.78.

References

1. Bell, C.E., Lewis, M.: The Lac repressor: A second generation of structural and functional studies. Curr. Opin. Struct. Biol. 11, 19–25 (2001)
2. Matthews, K.S.: The whole lactose repressor. Science 271, 1245–1246 (1996)
3. Muller-Hill, B.: Some repressors of bacterial transcription. Curr. Opin. Microbiol. 1, 145–151 (1998)
4. Pace, H.C., Kercher, M.A., Lu, P., Markiewicz, P., Miller, J.H., Chang, G., Lewis, M.: Lac repressor genetic map in real space. Trends Biochem. Sci. 22, 334–339 (1997)
5. Lewis, M.: The lac repressor. C.R. Biol. 328, 521–548 (2005)
6. Muller-Hill, B.: Suppressible regulator constitutive mutants of the lactose system in Escherichia coli. J. Mol. Biol. 15, 374–376 (1966)
7. Muller, J., Barker, A., Oehler, S., Muller-Hill, B.: Dimeric lac repressors exhibit phase-dependent co-operativity. J. Mol. Biol. 284, 851–857 (1998)
8. Pfahl, M., Stockter, C., Gronenborn, B.: Genetic analysis of the active sites of lac repressor. Genetics 76, 669–679 (1974)
9. Platt, T., Files, J.G., Weber, K.: Lac repressor. Specific proteolytic destruction of the NH 2 -terminal region and loss of the deoxyribonucleic acid-binding activity. J. Biol. Chem. 248, 110–121 (1973)
10. Schmitz, A., Schmeissner, U., Miller, J.H.: Mutations affecting the quaternary structure of the lac repressor. J. Biol. Chem. 251, 3359–3366 (1976)
11. Alberti, S., Oehler, S., von Bergmann, B., Kramer, H., Muller-Hill, B.: Dimer-to-tetramer assembly of Lac repressor involves a leucine heptad repeat. New Biol. 3, 57–62 (1991)
12. Alberti, S., Oehler, S., von Bergmann, B., Muller-Hill, B.: Genetic analysis of the leucine heptad repeats of Lac repressor. Embo. J. 12, 3227–3236 (1993)
13. Suckow, J., Markiewicz, P., Kleina, L.G., Miller, J., Kisters-Woike, B., Muller-Hill, B.: Genetic studies of the Lac repressor. XV. J. Mol. Biol. 261, 509–523 (1996)
14. Markiewicz, P., Kleina, L.G., Cruz, C., Ehret, S., Miller, J.H.: Genetic studies of the lac repressor XIV. J. Mol. Biol. 240, 421–433 (1994)
15. Kleina, L.G., Miller, J.H.: Genetic studies of the lac repressor XIII. J. Mol. Biol. 212, 295–318 (1990)

16. Vaisman, I.I., Tropsha, A., Zheng, W.: Compositional preferences in quadruplets of nearest neighbor residues in protein structures: Statistical geometry analysis. In: Proceedings of the IEEE Symposia on Intelligence and Systems, pp. 163–168 (1998)
17. Singh, R.K., Tropsha, A., Vaisman, I.I.: Delaunay tessellation of proteins: Four body nearest-neighbor propensities of amino acid residues. J. Comput. Biol. 3, 213–221 (1996)
18. Masso, M., Lu, Z., Vaisman, I.I.: Computational mutagenesis studies of protein structure-function correlations. Proteins 64, 234–245 (2006)
19. Verzilli, C.J., Whittaker, J.C., Stallard, N., Chasman, D.: A hierarchical Bayesian model for predicting the functional consequences of amino acid polymorphisms. Applied Statistics 54, 191–206 (2005)
20. Krishnan, V.G., Westhead, D.R.: A comparative study of machine-learning methods to predict the effects of single nucleotide polymorphisms on protein function. Bioinformatics 19, 2199–2209 (2003)
21. Karchin, R., Kelly, L., Sali, A.: Improving functional annotation of non-synonomous SNPs with information theory. Pac. Symp. Biocomput., 397–408 (2005)
22. Ng, P.C., Henikoff, S.: SIFT: Predicting amino acid changes that affect protein function. Nucleic Acids Res. 31, 3812–3814 (2003)
23. Berman, H.M., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T.N., Weissig, H., Shindyalov, I.N., Bourne, P.E.: The Protein Data Bank. Nucleic Acids Res. 28, 235–242 (2000)
24. Barber, C.B., Dobkin, D.P., Huhdanpaa, H.T.: The quickhull algorithm for convex hulls. ACM Transactions on Mathematical Software 22, 469–483 (1996)
25. Bell, C.E., Lewis, M.: A closer view of the conformation of the Lac repressor bound to operator. Nat. Struct. Biol. 7, 209–214 (2000)
26. Masso, M., Vaisman, I.I.: Comprehensive mutagenesis of HIV-1 protease: A computational geometry approach. Biochem. Biophys. Res. Commun. 305, 322–326 (2003)
27. Quinlan, R.: C4.5: Programs for Machine Learning, San Mateo, CA. Morgan Kaufman Publishers, San Francisco (1993)
28. Frank, E., Hall, M., Trigg, L., Holmes, G., Witten, I.H.: Data mining in bioinformatics using Weka. Bioinformatics 20, 2479–2481 (2004)
29. Fawcett, T.: ROC graphs: Notes and practical considerations for researchers. HPL-2003-4. Hewlett-Packard Labs, Palo Alto (2003)
30. Hanley, J.A., McNeil, B.J.: The meaning and use of the area under a receiver operating characteristic (ROC) curve. Radiology 143, 29–36 (1982)
31. Provost, F., Domingos, P.: Well-trained PETs. CeDER Technical Report IS-00-04. Stern School of Business, New York University, New York (2001)
32. Dayhoff, M.O., Schwartz, R.M., Orcut, B.C. (eds.): A model for evolutionary change in proteins, Washington D.C. National Biomedical Research Foundation, vol. 5 (1978)
33. Chasman, D., Adams, R.M.: Predicting the functional consequences of non-synonymous single nucleotide polymorphisms: Structure-based assessment of amino acid variation. J. Mol. Biol. 307, 683–706 (2001)
34. Wrobel, J.A., Chao, S.F., Conrad, M.J., Merker, J.D., Swanstrom, R., Pielak, G.J., Hutchison, C.A.: A genetic approach for identifying critical residues in the fingers and palm subdomains of HIV-1 reverse transcriptase. Proc. Natl. Acad. Sci. U.S.A. 95, 638–645 (1998)
35. Kyte, J., Doolittle, R.F.: A simple method for displaying the hydropathic character of a protein. J. Mol. Biol. 157, 105–132 (1982)

Evaluating Genetic Algorithms in Protein-Ligand Docking

Rafael Ördög and Vince Grolmusz

Protein Information Technology Group, Eötvös University, Budapest and
Uratim Ltd., Nyíregyháza, Hungary
{devill, grolmusz}@cs.elte.hu

Abstract. *In silico* protein-ligand docking is a basic problem in pharmaceutics and bio-informatics research. One of the very few protein-ligand docking software with available source is the Autodock 3.05 of the Scripps Research Institute. Autodock 3.05 uses a Lamarckian genetic algorithm for global optimization with a Solis-Wets local search strategy. In this work we evaluate the convergence speed and the deviation properties of the solution produced by Autodock with diverse parameter settings. We conclude that the docking energies found by the genetic algorithm have uncomfortably large deviations. We also suggest a method for considerably decreasing the deviation while the number of evaluations will not be increased.

1 Introduction

In silico protein-ligand docking methods are becoming more and more important in searching for new drug candidate molecules because of their speed, economy and increasing reliability. Acquiring one compound (or ligand) for wet-laboratory testing from compound manufacturers costs around $ 100, consequently, without counting the costs of labor, the additional reagents and the protein production, in vitro verifying of the binding of one million compounds against one protein may cost $ 100 million. *In silico* simulation of the binding by docking methods costs only a small fraction of that amount, and can be completed in 1-2 weeks on a computer cluster of moderate size. The key ingredient of the *in silico* docking is the docking algorithm. Each docking algorithm optimizes some scoring function for finding the best location and conformation of the ligand molecule near to the surface of the protein. As an input, one must use the three-dimensional coordinates of the protein (usually taken from the Protein Data Bank) and the ligand molecule. As the output, a docking algorithm returns one or more docked conformation of the ligand and the protein, and the corresponding values of the scoring function.

1.1 The AutoDock Docking Software

One of the most widely known docking software with acquirable source code is the AutoDock 3.05 of the Scripps Research Institute [1]. Note, that the source

I. Măndoiu, R. Sunderraman, and A. Zelikovsky (Eds.): ISBRA 2008, LNBI 4983, pp. 402–413, 2008.

code of such popular docking software as Dock[2], Gold[3], Fred[4], FlexX[5] and many others are not available at all. In AutoDock 3.05 the scoring function is the estimated docking energy of the ligand to the protein. The best docking is the one with the smallest energy. The derivation of the empirical binding free energy function is described in [1], along with a brief historical review of the topic. It is easy to see that the three-dimensional position of any rigid molecule can be described by 6 real variables (three Euler angles plus the position of one of its atoms). If ℓ torsion axes are also allowed, then the scoring function should be optimized in a real space of dimension $6 + \ell$.

To speed up evaluation of the energy function AutoDock uses a grid-based approach. First, the protein is preprocessed by a program called AutoGrid: probe atoms and charges are placed at grid points around the protein, and the energy function is calculated and stored for latter use. After that AutoDock uses trilinear interpolation between grid-points to determine energy terms for each the atoms of the ligand separately, and then sums them up to calculate the energy of the conformation. The strategy is to minimize the energy function above in the $6 + \ell$ dimensional real space, where each point of the space corresponds to a conformation of the ligand. The optimization procedure clearly distinguishes a *local optimization* and a *global optimization* strategy. The aim of the local minimization strategies (local search) is to find a local minimum of the function in the neighborhood of the starting point. The aim of the global minimization strategy is to find the minimum of the function on the whole domain.

We need to mention that although there exist a good number of local search techniques finding local optima reliably for any continuous function, the *No free lunch theorem* [6] states that it is impossible to find a general purpose algorithm for global optimization, that performs equally well on all functions. Hence it is not trivial – if possible at all – to choose a global strategy, that suits a class of functions well. In order to circumvent this problem, one need to utilize as much information about the function as possible.

The first versions of AutoDock were using Simulated Annealing (SA) as global optimization strategy [1]. Further investigations [7] showed, that Genetic Algorithms with Local Search (GA-LS) – discussed in the next section – outperform the SA strategy. AutoDock Version 3 and later has both SA and GA-LS implemented. Neither for SA nor for GA-LS convergence results are known in the case of the energy function above.

1.2 Genetic Algorithms

Algorithms - performing function optimization - based on the principles of Darwinian Evolution, are called Evolutionary Algorithms. These algorithms maintain a collection of possible solutions (individuals) and select certain individuals for further processing depending on their fitness, i.e., the function value at the point represented by the individual. The most widely used of these algorithms are the Genetic Algorithms. In this section we sketch the basic properties of Genetic Algorithms. A more detailed description is given by Whitley [8].

An *Individual* is a point in the search space, and its *Genotype* is the string of numbers (or vector) that describes it. The *Phenotype* is the collection of attributes of the individual, and its *Fitness* is the function value corresponding to the individual. A *Population* is simply a collection of individuals. The algorithm first selects a population of (usually random) individuals that form the first *Generation*, then enters a cycle of deriving the n^{th} generation of individuals from the preceding. Every generation has the same fixed size. A cycle of the algorithm performs a *selection*, and applies *variation operators* to the individuals in the current population. We perform a *selection* by randomly choosing each individual of the

Genetic Algorithms with Local Search. (GA-LS) introduce a further unary variation operator, that is usually not considered a mutation operator, and are typically applied at the end of a cycle. This variation operator is a type of *Local Search*, typically Solis & Wets random local search.

The GA-LS implementation in AutoDock. To initialize the first generation, AutoDock chooses a population randomly. Population size is a parameter that is set to 50 as default. Then it enters the generation-deriving process, and starts the global strategy. Selection is either proportional, or a probabilistic binary tournament. We applied proportional choices for our experiments. Before the crossover step, individuals are being permuted, and then subsequent $2i^{th}$ and $(2i+1)^{th}$ individuals are being applied the crossover operator with a given probability, that is $\frac{4}{5}$ by default. AutoDock is using either one or two point block crossover (latter being our choice) where the blocks are real values for translation, rotation, and torsions. Cauchy mutation is being applied with default parameters mean 0, variance 1, and probability of applying the mutation operator is $\frac{1}{50}$. Elitism was set to keep the best individual alive. As a default every individual of a generation is applied a local search with the probability $\frac{2}{3}$, maximal number of steps is 300. All further parameters are left as default in AutoDock.

1.3 Previous Work

Thomsen's remarkable paper [9] analyzes genetic algorithm parameterization in the AutoDock 3.05 software. Six protein-ligand complexes were chosen for the experiments, and the effect of different population sizes, mutation operators, recombination operators, different local search strategies were tested for these six complexes. Based on the findings, in [9] a new algorithm, called DockEA was introduced and compared with the original AutoDock solutions. The results were quite different for different complexes: for some of the complexes it was not too difficult to find the near-optimum solutions, and for some others it was a more challenging task. Note also, that the resolution of the complexes examined were in the range 1.63-3.1 Å. The termination criterion was set to 250,000 evaluations and each experiments were repeated with 30 random seeds. We, in contrast, docked 48 randomly chosen ligands to the same protein: the

Mycobacterium tuberculosis dUTPase protein with PDB code of 2PY4, and each docking was repeated 100 times with different random seeds. The main point of our analysis was to find the number of evaluations which already gives reliable results, but it is as low as possible to facilitate screening large ligand libraries. Consequently, we perform our experiments with higher evaluations upper bound than 250,000; in some cases the upper bound was set to 2 million.

1.4 Concept of Our Analysis

Clearly, an "idealistic" randomized algorithm that performs automated docking need to satisfy some natural requirements. Our main motive was to optimize a database screening task with a single protein against two million ligand molecules, without changing the energy function of AutoDock. In what follows we list our requirements and give methods to quantitatively evaluate them.

Capability of finding optimum: This requirement means that with high enough probability the algorithm approximates the minimum of the function with a small enough error. Evaluating this requirement is difficult, since we do not have reliable measurement data on the values of the energy-minima for multiple ligands in question. To circumvent these difficulties, we compared different variants of the GA algorithm based on a fixed number of runs each with the same energy function.

Mean approximates best run - (Low deviation). It is reasonable to expect that a result, yielded by a given run of the algorithm is generally close to the optimum.

Consistency of the above quantities means that if the number of evaluations is increased, then the discovered optima get ever closer to the real optimum, and the deviation drops as well.

Order of hits. When thousands or millions of different ligands are docked against the same protein, identifying the order of the ligands, sorted by the docking energies, is sometimes much more important than the absolute values of the docking energies themselves.

To see the progress made by different algorithms we created run logs similar to the ones in Hart's thesis [7]. After every evaluation we checked if a new optimum was discovered, and recorded the number of evaluations and the new optima. On plots like the one on Figure 1 we can see number of evaluations on the horizontal axes, and the approximation of minimal energy in kcal/mol calculated after the given number of evaluations by different runs of the algorithm. Note that the most time-consuming step is the evaluation of the energy function, so the number of evaluations is proportional to the running time of the algorithm.

2 Methods

Our test-runs were performed on a cluster of 48 PC's equipped with 2.40GHz Intel Celeron^{TM} processors and 256 MB RAM. We have randomly chosen 48 molecules from the ZINC database [10], and docked them to the same *Mycobacterium tuberculosis* dUTPase molecule, with PDB code 2PY4, as a protein target. The algorithm ran with 100 different random seeds for every ligand to determine

deviation and seed dependency. The diagrams derived from the run logs by the procedure described at the end of the previous section were similar to Figure 1, and in what follows we will describe the common properties we have found:

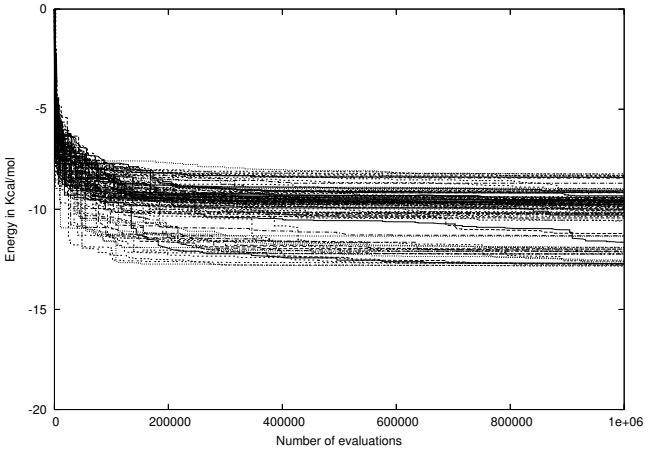


Fig. 1. Evolution of discovered minima over 100 runs with different seeds for ligand with ZINC code ZINC01208228

- One should note that near to the beginning the algorithm comes across unreasonably high positive energies due to collisions, but in several thousand steps it reaches negative values.
- Shortly after reaching negative energies, the best of the 100 runs finds values close to the minimum. For example, on Figure 1, the minimal run changes less than 0.5 kcal/mol after 250,000 steps. For the 71% of the 48 ligands we were testing this changed less than 0.5 kcal/mol after 250,000 evaluations, and 88% changed less than 1 kcal/mol. The worst case was 2 kcal/mol.
- As a generalization of the above observation one can see on Figure 1, that even after 10,000,000 evaluations, most of the runs stay close to the value reached after 250,000 evaluations. Note, however, that there are seeds producing large jumps at random positions (e.g., on Figure 1, just after 400,000 evaluations, where one descends from -10 kcal/mol to -12 kcal/mol.)

  Figure 3 shows 4,800 different runs (100 for each ligands), ordered by the amount of decrease of the docking energy found in kcal/mol from evaluation 250,000 to evaluation 10,000,000. The 50% of all runs to change less than 0.5 kcal/mol, and only 25% of the runs decreased more than 1 kcal/mol in this interval.
- Motivated from our analysis above, we define the "high confidence" interval of energies by the following properties:
  - Every discovered minimum is located in this interval independently of seed.
  - It is the smallest interval that satisfies the above.

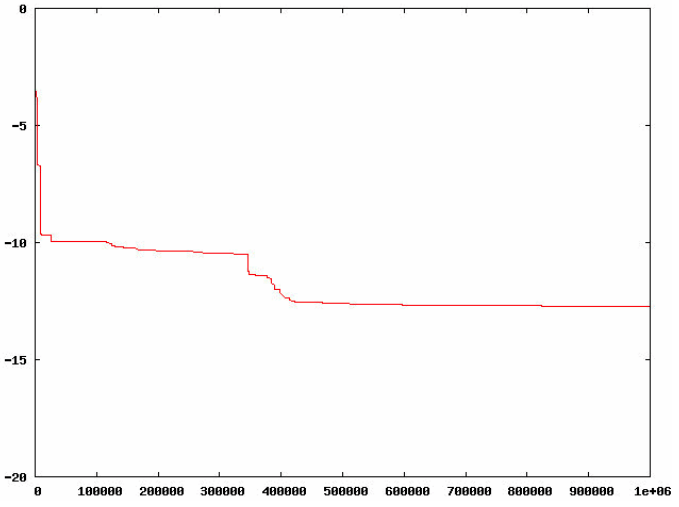


Fig. 2. Evolution of discovered minima with a seed for ligand with ZINC code ZINC01208228. Notice the sudden fall just before 350 000 evaluations.

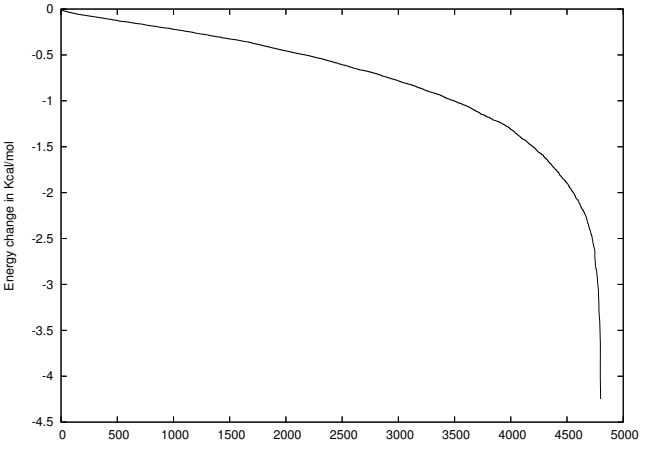


Fig. 3. 4,800 different runs ordered by amount of decreasing in kcal/mol from the 250,000th evaluation to the 10,000,000th evaluation

- It is worth mentioning that high confidence intervals can be approximated by considering only a small number of runs, as it can be seen on Figure 1. Figure 5 represents the "high confidence" intervals for different ligands with vertical bars, and the mean of 100 runs with a dot on the corresponding bar.
- Based on our 100 run approximations:
  - Usually the minima gained in 100 runs cover the high confidence interval almost uniformly. Figure 1 is a counterexample of this phenomenon, since

it has a gap between -10 and -12 kcal/mol. From the 48 ligands another 3 ligand also have similar gaps.
  - After 250,000 evaluations the endpoints of the high confidence intervals will not be changed much. More exactly, for the 71% of the 48 ligands we were testing the interval changed less than 0.5 kcal/mol after 250,000 evaluations, and for 88% it changed less than 1 kcal/mol. The worst case was 4 kcal/mol, but the second worst was only 1.5 kcal/mol.
  - The approximate size of the high confidence intervals is between 2 and 5 kcal/mol. (For example the energies for the ligand with ZINC code ZINC02958278 were between -11 and -7 kcal/mol.)
- Neither the standard deviation, nor the mean of 100 runs show consistent decrease after 250,000 evaluations. (See Figure 4.)

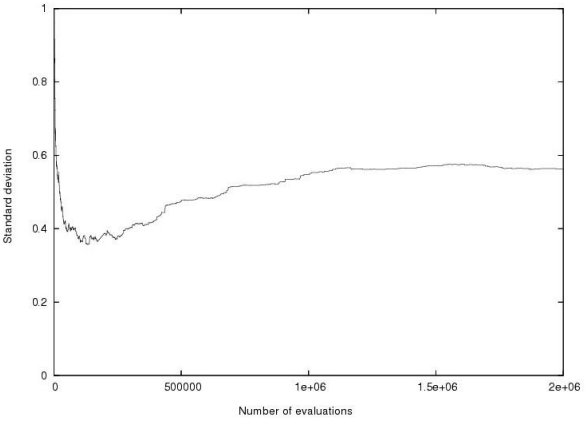


Fig. 4. Increasing deviation of different runs on the ligand with ZINC code ZINC01106466

One of the main conclusions of ours is that if ligands are characterized by the minimal energies discovered over 100 different runs, the difference between the best and worst ligands was less than 6.5 kcal/mol for this 48 random choices, i.e., we have got values between -13.5 and -7. Note that the length of that interval is clearly comparable with the lengths of high confidence intervals.

On Figure 5 the dots on the intervals indicate the average of runs. One can see that they are usually in the middle of the confidence interval, hence we can not expect a single randomly chosen run to be at least near to the optimum. What's more disturbing, if we accept the rather optimistic assumption of having chosen the "average run" for all the ligands by luck, we are still far from a relevant order, as — due to the large variance in the size of intervals — there seems to be no relationship between the minimal and average runs.

Another surprising result was, that deviation sometimes was increased after a number of evaluations instead of the anticipated decreasing. (See figure 4.) This pathological behavior arises when there is an easy to discover optimum,

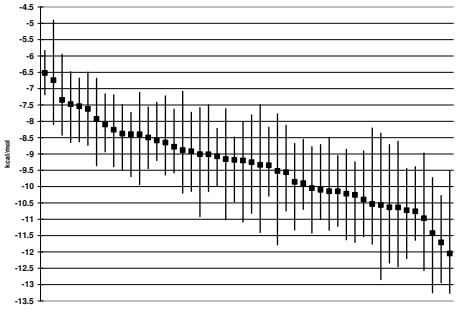


Fig. 5. High confidence intervals for 48 ligand molecules after 2,000,000 evaluations

and most of the runs find it within the first few thousand steps, and afterward they start to explore better solutions at random positions in time.

3 Results and Discussion

3.1 Multi-run Tests

In the case of high deviation it is a quite common approach to take the average, or the minimum of multiple samples. As we pointed out in the previous section averages are not relevant, consequently we tried multi-run algorithms with common minima, i.e., an n-run strategy with k evaluations would run the algorithm k times through $\frac{k}{n}$ evaluations, and take the minimum of the results. Our test described below showed that optimal n for a given k is highly protein-ligand dependent, though we believe that for a group of similar ligands and fixed protein it might be possible to find a common n. If ligand classes would be large enough it might be possible to save time by preprocessing ligand classes for every protein in order to identify the optimal n. The question whether is this possible remains open.

Our tests were performed as follows: we ran the algorithm 100 times, and divided it into 10 equal-sized group. For the average n run strategy the minimum of the first n out of every group was taken, and after that the mean of them to get an average n run strategy.

It is obvious, that for small number of evaluations any multi-run strategy is no match for the one-run strategy, and for significantly large number of evaluations in multi-run strategies will overcome the one-run strategy. (The latter is obvious considering the results of the previous Section, i.e., most runs get stuck at a point.) The question remains when does an n run strategy overcome an k run strategy, and more importantly which is the best for a given number of evaluations.

On Figure 6 the average of the n-run strategies plotted for ZINC entry ZINC00342090. In this example, the 3-run strategy is the first to overcome the one-run strategy at approximately 700,000 evaluations, but there is not enough

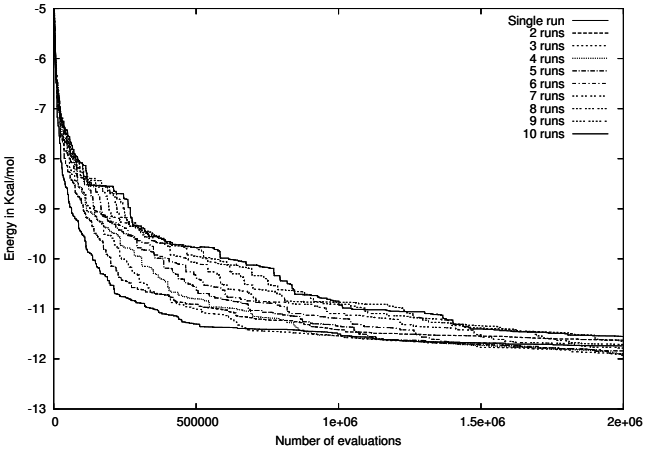


Fig. 6. Different multi run strategies for ZINC entry ZINC00342090

difference between them to ensure that the 3 run strategy is definitely better. Other ligand showed examples where the 1 run strategy turned out to be far better than any other strategy even at 2,000,000 evaluations, and there were examples, where the 2 and 3 run strategies overcame the single run version pretty well.

3.2 Modified GA-LS Algorithms

The choice of the initial population is an important phase of Genetic Algorithms. Its optimization seems to be neglected by the literature. This is not surprising in a certain sense, as the algorithm usually leaves those individuals behind at a quite early stage, hence a possible bad initial choice seems to have only a negligible effect on later generations. Our investigations show that this intuition is far from being correct. In the case of a function with such high complexity as our energy function, the choice of initial population can have strong positive effect on the speed of convergence to the actual optimum, by more or less restricting the search to interesting areas.

On figure 7 one can observe the average of 100 runs with different initialization strategies, for ZINC entry ZINC01208228. Again horizontal axes is number of evaluations, while vertical axes is energy in kcal/mol. Different strategies are described below:

Rigid start. In the ZINC database [10], ligands are stored in a conformation with minimal internal energy. Hence it is a natural idea to fix the torsions when choosing the individuals randomly. Our aim was to reduce the first "many collisions" phase of the algorithm, but the diversity of the population - as we expected - turned out to be too small, and that slowed down the algorithm significantly.

First population selected from a larger random population is another natural idea. To understand the behavior of random individuals we have created plots

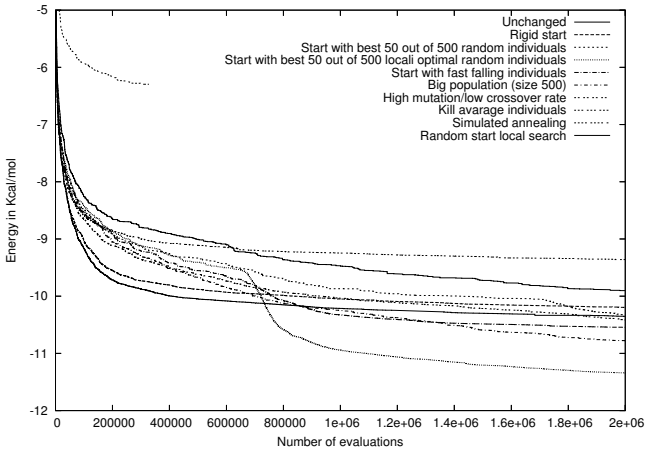


Fig. 7. Comparing the modified algorithms (x-axis) by average run and confidence interval for ZINC entry ZINC00342090 in kcal/mol (y-axis)

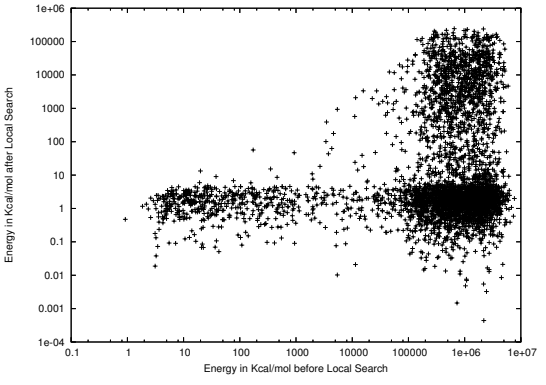


Fig. 8. Energies of random individuals, and of the nearest local optima found by local search

like the one on Figure 8 that show the energies (horizontal axes) for 5,000 random individuals, and the energies for the nearest local optimum found by local search (vertical axes). One can see that points form 3 classes, formed by low energy individuals, high energy individuals near low local optimum, and high energy individuals near high local optimum.

Selecting the best 50 individuals out of 200,000 turns out to perform more or less the same way as the unchanged algorithm. Depending on the ligand it can perform better or worse just as many times.

Selecting the best after local search with 20,000 being the number of local searched random individuals, and 50 the population size. As one can check on the example Figure 7, this approach turns out to be the best from the ones inspected by us. First it works its way through the local search phase, and then

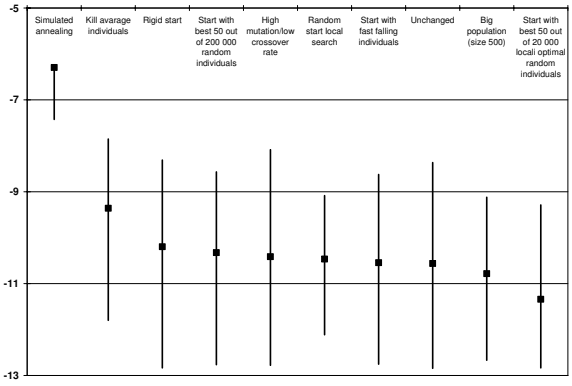


Fig. 9. Comparing the modified algorithms by high confidence interval after $2,000,000$ evaluations for ZINC entry ZINC00342090

soon after starting the genetic algorithm the average of the runs makes a sudden fall. At this point it outperforms all other approaches for most of the ligands.

Individuals changed by local search most dramatically seemed to be interesting, partly because they form a separate cluster (Figure 8), and partly because according to our notion optimal bindings are situated on the surface of the protein, hence changing it a little causes collisions, i.e., high energy conformations.

Non-initialization based modifications included higher population size of 500, modified mutation $\left(\frac{4}{5}\right)$ and crossover $\left(\frac{1}{5}\right)$ rates, and a modified version of proportional selection. The last modification aimed to exploit the remark mentioned earlier about optima being near to high energy conformations.

4 Conclusions

We performed an in-depth analysis of the settings of the GA-LS algorithm implemented in AutoDock 3.05, and concluded, that they tend to have high deviations when applied to energy functions of docking problems. The results were obtained using one single protein as a docking target; this fact yields consistency in the results, but may also bound the the generality of our results.

Consequently, one can not expect to reach exact energy bounds with this technique, nor to find a relevant order of ligands according to their bonding energies within a low number of evaluations. Multi-run strategies can help, but they are highly dependent on the protein-ligand pair in question.

Initialization can have a major effect on the speed of convergence. The best algorithm examined in this article was choosing the best 50 out of $20,000$ local searched individuals. The first phase of this algorithm is actually a random start local search, but after the genetic algorithm is started the average of runs makes a sudden fall, and outperforms all other variants we have examined. Figure 9 shows that this variant performs better than others in average.

Acknowledgment. The authors acknowledge the partial support of an OTKA grant NK 67867 and NKTH project "TB_INTER", and invaluable help from Zoltan Szabadka.

References

[1] Morris, G.M., Goodsell, D.S., Halliday, R.S., Huey, R., Hart, W.E., Belew, R.K., Olson, A.J.: Automated docking using a lamarckian genetic algorithm and an empirical binding free energy function. Journal of Computational Chemistry 19(14), 1639–1662 (1998)

[2] Ewing, T., Makino, S., Skillman, A., Kuntz, I.: Dock 4.0: Search strategies for automated molecular docking of flexible molecule databases. J. Comput. Aided. Mol. Des. 15(5), 411–428 (2001)

[3] Verdonk, M., Cole, J., Hartshorn, M., Murray, C., Taylor, R.: Improved protein-ligand docking using gold. Proteins 52(4), 609–623 (2003)

[4] Schulz-Gasch, T., Stahl, M.: Binding site characteristics in structure-based virtual screening: evaluation of current docking tools. Journal of Molecular Modeling 9(1), 47–57 (2003)

[5] Rarey, M., Kramer, B., Lengauer, T., Klebe, G.: A fast flexible docking method using an incremental construction algorithm. J. Mol. Biol. 261(3), 470–489 (1996)

[6] Wolpert, D.H., Macready, W.G.: No free lunch theorems for imization. IEEE Transactions on Evolutionary Computation (1997)

[7] Hart, W.E.: Adaptive Global imization with Local Search. PhD thesis, University of California, San Diego (1994)

[8] Whitley, D.: A genetic algorithm tutorial. Statistics and Computing 4, 65–85 (1994)

[9] Thomsen, R.: Flexible ligand docking using evolutionary algorithms. Investigating the effects of variation operators and local-search hybrids. BioSystems 72(1–2), 57–73 (2003)

[10] Irwin, J.J., Shoichet, B.K.: A free database of commercially available compounds for virtual screening. J. Chem. Inf. Comput. Sci. 45(1), 177–182 (2005)

A Hidden Markov Model Approach for Prediction of Genomic Alterations from Gene Expression Profiling

Huimin Geng[1,2], Hesham H. Ali[1], and Wing C. Chan[2]

[1] Department of Computer Science, University of Nebraska at Omaha, Omaha, NE 68182
[2] Department of Pathology and Microbiology, University of Nebraska Medical Center, Omaha, NE 68198
{hgeng, hali}@mail.unomaha.edu, jchan@unmc.edu

Abstract. The mRNA transcript changes detected by Gene Expression Profiling (GEP) have been found to be correlated with corresponding DNA copy number variations detected by Comparative Genomic Hybridization (CGH). This correlation, together with the availability of genome-wide, high-density GEP arrays, supports that it is possible to predict genomic alterations from GEP data in tumors. In this paper, we proposed a hidden Markov model-based CGH predictor, HMM_CGH, which was trained in the light of the paired experimental GEP and CGH data on a sufficient number of cases, and then applied to new cases for the prediction of chromosomal gains and losses from their GEP data. The HMM_CGH predictor, taking advantage of the rich GEP data already available to derive genomic alterations, could enhance the detection of genetic abnormalities in tumors. The results from the analysis of lymphoid malignancies validated the model with 80% sensitivity, 90% specificity and 90% accuracy in predicting both gains and losses.

Keywords: Gene Expression Profiling (GEP), Comparative Genomic Hybridization (CGH), Hidden Markov Model (HMM), Genomic Alterations.

1 Introduction

Gene expression profiling (GEP) and comparative genomic hybridization (CGH) are two important genome-wide microarray techniques to study tumorigenesis. Specifically, GEP measures mRNA expression levels, which has been widely used to identify genes differentially expressed between tumor and normal samples or between cancer subtypes [1, 2]. CGH is a molecular cytogenetic technique to detect DNA copy number variations, having been widely used in defining the putative chromosomal regions involved in tumor progression [3-6]. Both CGH and GEP rely on hybridization of tumor samples to chips and image extraction through high resolution scanners.

Genetic alterations are key causes of tumorigenesis. The identification of genetic alterations (usually through CGH experiments) would provide important insights into the mechanisms of tumorogenesis. But in the post-genomic era, the majority of the studies in tumor biology focus on GEP but not CGH due to limitations in resolution. As a result, a substantial amount of GEP data have been accumulated in the last decade and made publicly available, but few CGH studies have been done with a

I. Măndoiu, R. Sunderraman, and A. Zelikovsky (Eds.): ISBRA 2008, LNBI 4983, pp. 414–425, 2008.

large series of cases. Therefore, it is advantageous to derive genomic alterations from GEP data, which are already available, but not performing actual CGH experiments. In addition, high resolution array CGH is expensive and labor intensive. It requires a separate DNA extraction from tumor biopsies. It is usually very difficult, if not impossible, for an investigative group to collect a large series of tumor specimens, which have GEP performed by other groups, for a new CGH study.

The advanced array techniques of genome-wide GEP and high-resolution array CGH enabled the direct correlation of gene expression changes and copy number variations on gene level throughout the genome. It has been shown by many studies that the mRNA transcript changes detected by GEP correlate to the corresponding DNA copy number variations detected by CGH, as in breast cancer [7-12], prostate cancer [13], leukemia [14], gastric cancer [15], sarcoma [16] and yeast mutants [17]. Our recent studies have also revealed the substantial correlation of GEP and CGH in lymphoid malignancies, such as diffuse large B-cell lymphoma [18], mantle cell lymphoma [19] and natural killer-cell lymphoma [20]. From the above studies, the expression of 30-50% of the genes present in the aberrant regions showed an association with the corresponding gains or losses. The association between GEP and CGH laid the biological foundation for predicting genomic alterations from GEP.

On the other hand, the advances in GEP arrays provide the technical foundation for the prediction. For example, Affymetrix Human Genome U133 (HG-U133) Set (A and B) contains about 45,000 probesets interrogating short regions of the human genome, representing more than 39,000 transcripts. The upgraded HG-U133 Plus 2.0 contains even more probesets (over 54,000). With about 19,000 genes measured on the HG-U133 arrays, the majority of the predicted 20,000 to 25,000 human genes are covered [21]. In addition, the concerns about reliability issues of microarray measurements has been recently addressed by the MicroArray Quality Control (MAQC) projects, which finds that microarray data is reasonably reproducible within and across different microarray platforms, that consensus on data analysis appears to be attainable, and that microarray technologies are sufficiently reliable to be used for clinical and regulatory purposes [22].

In this study, we proposed a novel computational method using hidden Markov models (HMM) to predict chromosomal gains and losses based on GEP data, called HMM_CGH predictor. It takes advantage of rich GEP data and could significantly improve the identification of genomic abnormalities from both scientific and economical point of view. The rest of the paper is organized as follows. In section 2, we describe the HMM_CGH model, including model structure, training procedure and prediction procedure. In section 3, we illustrate the flowchart of the overall process performing HMM_CGH model on real tumor datasets and introduce methods and criteria to validate HMM_CGH at both gene and cytoband level. In section 4, we provide the model performance by applying it to the analysis of lymphoid malignancies. In section 5, we conclude that HMM_CGH predictor is a powerful tool that may significantly enhance the data analysis of GEP in cancer research.

2 Methodology of HMM_CGH Predictor

HMMs are well developed statistical models. They have been widely and successfully used in capturing information buried in biological sequences, such as in finding

protein secondary structure, CpG islands and families of related DNA or protein sequences. The common problem formulation using HMM in biological sequences is, "Given a sequence of symbols as observations, predict the hidden state for each symbol along the sequence." Applying to our HMM_CGH problem, the observations are a sequence of symbols from GEP— *H*, *L* and *M* for high, low and medium expression, respectively; the labels for a hidden state are CGH status— "+", "-" and "*o*" for gain, loss and normal (unchanged) of chromosomal regions, respectively.

2.1 HMM_CGH Model Structure

A HMM describes a doubly-embedded stochastic process with one observable process $\{O_i\}$ and one hidden process $\{H_i\}$. In our HMM_CGH problem, the observable process is a sequence of symbols $\{x_i\}$ (x_i = *H*, *L* or *M*), and the hidden process is the underlying state path $\{\pi_i\}$ (π_i = "+", "-" or "*o*"). In a HMM, the state is not directly visible, but variables influenced by the state are visible. The model parameters for a HMM come from two categories: 1) state transition probabilities, $a_{kl} = P(\pi_i = l \mid \pi_{i-1} = k)$, which is a probability from state *k* to state *l*; 2) emission probabilities, $e_l(b) = P(x_i = b \mid \pi_i = l)$, which is a probability distribution over all the possible output symbols *b* for each state *l*.

Fig. 1 illustrates the HMM_CGH model in a Bayesian network representation, where the shaded S_1, S_2, …, S_n represent hidden state variables and the visible E_1, E_2, …, E_n represent the observations for the variables. The observable emission space consists of three symbols {*H*, *L*, *M*} and the hidden state space consists of nine states $\{H_+, L_+, M_+, H_-, L_-, M_-, H_o, L_o, M_o\}$, where H_+, L_+ and M_+ emit *H*, *L* and *M* in "+" region, H_-, L_- and M_- emit *H*, *L* and *M* in "-" region, and H_o, L_o, M_o emit *H*, *L* and *M* in "*o*" region.

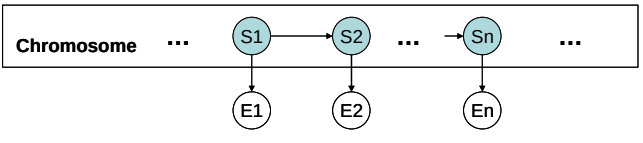


Fig. 1. HMM_CGH model presented as a Bayesian network. The hidden states variables are shaded as S_1, S_2, …, S_n. The observations for the variables are E_1, E_2, …, E_n. The observable emission space is {*H*, *L*, *M*}. The hidden states space is $\{H_+, L_+, M_+, H_-, L_-, M_-, H_o, L_o, M_o\}$.

Fig. 2 shows the HMM_CGH model presented by the state transition diagram. The model is a single chain that incorporates three Markov chains: sub-chain (+), sub-chain (-) and sub-chain (*o*). In each sub-chain, there is a complete set of transitions. The transitions among the three sub-chains are also allowed. This design makes it possible to identify gain and loss regions of variable length along a chromosome automatically by screening the entire chromosome without the fixed-window-size problem. It's obvious from the transition diagram that the transition probability parameters are composed of a 9x9 matrix (a combination of the pairing of nine states) and the emission probability parameters are composed of a 3x9 matrix (a combination of the pairing of nine states and three symbols).

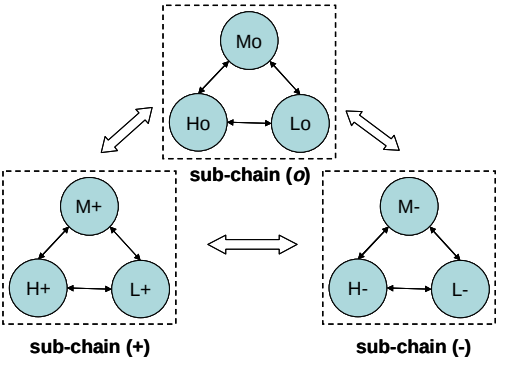


Fig. 2. State transition diagram of HMM_CGH model. The model is a single HMM chain integrating three Markov sub-chains: "+", "-" and "*o*". Each Markov sub-chain is presented as a collection of states with arrows between them representing the state transition. The state transitions among the three sub-chains are also allowed, as shown by the big arrows.

Fig. 3 shows the input and output of HMM_CGH. The input includes a fully specified HMM_CGH model, which would be obtained by model training, and a sequence of GEP observations. The output is a sequence of hidden states representing gain/loss/normal states, which is done through prediction.

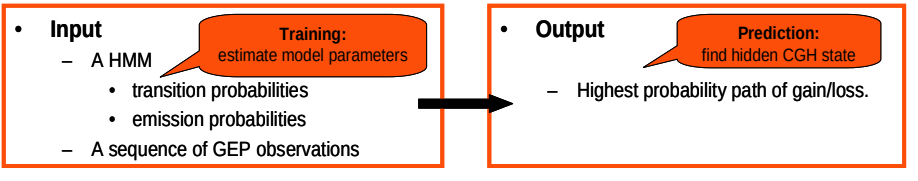


Fig. 3. Input and output of HMM_CGH. The model has two major parts: training and prediction.

2.2 HMM_CGH Training

In our model, emission parameters are deterministic, independent of training data, as shown in Eq. 1. Transition parameters can be estimated by Maximum Likelihood Estimation (MLE) as shown in Eq. 2, where a_{kl} are the transition probability and A_{kl} is the number of times that the state transition (k to l) happens in training data.

$$E_{3\times 9}=\begin{pmatrix} e_{H_+}(H)=1 & e_{H_-}(H)=1 & e_{H_o}(H)=1 & e_{L_+}(H)=0 & e_{L_-}(H)=0 & e_{Lo}(H)=0 & e_{M_+}(H)=0 & e_{M_-}(H)=0 & e_{M_o}(H)=0 \\ e_{H_+}(M)=0 & e_{H_-}(M)=0 & e_{H_o}(M)=0 & e_{L_+}(M)=0 & e_{L_-}(M)=0 & e_{Lo}(M)=0 & e_{M_+}(M)=1 & e_{M_-}(M)=1 & e_{M_o}(M)=1 \\ e_{H_+}(L)=0 & e_{H_-}(L)=0 & e_{H_o}(L)=0 & e_{L_+}(L)=1 & e_{L_-}(L)=1 & e_{Lo}(L)=1 & e_{M_+}(L)=0 & e_{M_-}(L)=0 & e_{M_o}(L)=0 \end{pmatrix} \quad (1)$$

$$a_{kl} = A_{kl} / (\sum_{l'} A_{kl'}) \quad (2)$$

2.3 HMM_CGH Prediction

Having the model parameters estimated, we can now use the fully specified model to estimate the hidden state paths for a new sequence of GEP observations. *Viterbi* algorithm [23] is used to decode hidden states. *Viterbi* algorithm is a dynamic programming algorithm. It estimates the state path by finding out the most likely one, $\pi^* = \arg\max_{\pi} P(x,\pi)$. Suppose the probability $v_k(i\text{-}1)$ of the most probable path ending in state k with observation $x_{i\text{-}1}$ is known for all the states k, then the probability corresponding to the observation x_i with the state l can be calculated as: $v_l(i) = e_l(x_i)\max_k(v_k(i-1)a_{kl})$. The entire path π^* can be found recursively.

Viterbi Algorithm

Initialization(i=0): $v_0(0)=1,\ v_k(0)=0$ for $k>0$.

Recursion(i=1…L): $v_l(i)=e_l(x_i)\max_k(v_k(i-1)a_{kl});\ ptr(l)=\arg\max_k(v_k(i-1)a_{kl})$.

Termination: $P(x,\pi^*)=\max_k(v_k(L)a_{k0});\ \pi_L^*=\arg\max_k(v_k(L)a_{k0})$.

Traceback(i=1…L): $\pi_{i-1}^*=ptr_i(\pi_i^*)$.

3 Validation of HMM_CGH Predictor

We tested the performance of HMM_CGH model using cross validation on real tumor cases. The scheme is shown in Fig. 4 with the following steps. (1) *Case splitting.* All the cases, associated with the paired GEP and CGH data, are split into two

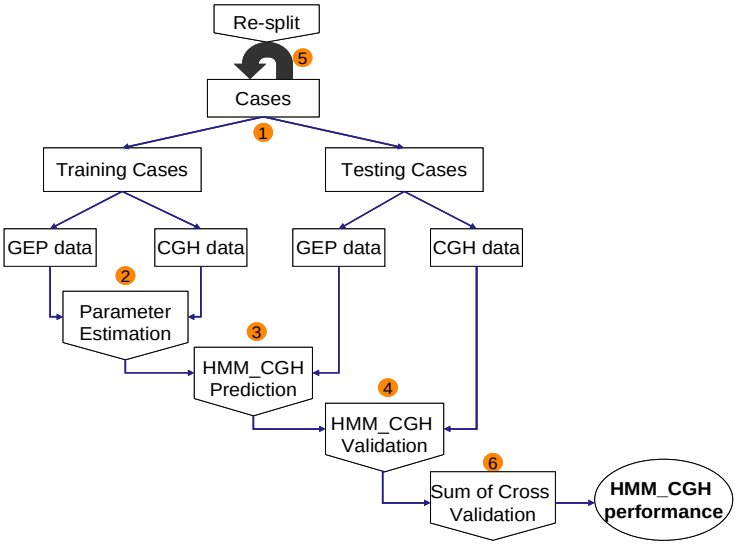


Fig. 4. The flowchart of performing and evaluating HMM_CGH model

sets— training and testing. In the training set, coupled GEP and CGH data are used for model construction; in the testing set, GEP are used for prediction while the corresponding CGH are used for validation. (2) *Parameter estimation.* The model parameters are estimated using both CGH and GEP data from the training dataset. (3) *Prediction.* The GEP data from the testing dataset are applied to the model to predict chromosomal gain and loss. (4) *Validation*. The predicted gain/loss regions are compared with those identified by experimental CGH on the same cases to evaluate prediction performance. (5) *Repeating*. The whole process is repeated by different splitting of training and testing cases. If leave-one-out cross validation (LOOCV) is used, which is the case in this study, the repeated splitting is performed n times (n is the size of the dataset), each time leaving one case out for validation. (6) *Overall performance*. After testing all the cases, we sum up the results of individual cases and calculate the overall model performance.

3.1 Gene-Level Validation

We first do probeset-by-probeset comparisons of the predicted outcome (HMM_CGH) with the "gold" standard (experimental CGH). Specifically, we count the number of probesets in each category of, true positive (*TP*), true negative (*TN*), false positive (*FP*) and false negative (*FN*), for each testing case. Then summing up those numbers for all the cases from LOOCV, we calculate sensitivity, specificity and accuracy (Eq. 3) for the overall model performance.

$$\begin{aligned} Sensitivity &= TP / (TP + FN) \\ Specificity &= TN / (TN + FP) \\ Accuracy &= (TP + TN) / (TP + TN + FP + FN) \end{aligned} \tag{3}$$

3.2 Cytoband-Level Validation

Since our CGH data used for validation is the conventional CGH, which has the resolution on cytobands, we further extend the comparison from probeset level to cytoband level by applying a smoothing algorithm after HMM_CGH prediction. Basically, we use a multinomial model to estimate the likelihood of a cytoband being a gain or loss as a whole, based on the probesets located in that cytoband.

Suppose a cytoband contains n probesets. We count the occurrences of probesets in gain, loss and normal states predicted by HMM_CGH, denoted by n_+, n_- and n_o (obviously, $n_+ + n_- + n_o = n$). The likelihood of observing n_+, n_- and n_o under certain hypothesis H is computed from the multinomial distribution as in Eq. 4. The null hypothesis H_0 is "the cytoband does not belong to a DNA gain/loss region". The alternative hypothesis H_1 is "the cytoband does belong to a DNA gain/loss region". The parameters under H_0 can be estimated from the whole genome as the background, and the parameters under H_1 can be estimated from that particular cytoband, as shown in Eq. 5. Finally, we give a probability score, the log-of-odd (LOD), to measure the likelihood of a cytoband being a gain or loss. LOD score is defined as the log base 10 of the likelihood ratio under the hypotheses of H_1 and H_0 in Eq. 6. The higher the LOD score, the more likely this cytoband is a genomic gain or loss.

$$L(n_+, n_-, n_o \mid H) = \frac{n!}{n_+!\, n_-!\, n_o!} \theta_+^{n_+} \theta_-^{n_-} \theta_o^{n_o},\ (n_+ + n_- + n_o = n) \tag{4}$$

$$\hat{\theta}_\nabla = {}^{n_\nabla}\!/_{n},\ \nabla = \text{'+', '-', or 'o'} \tag{5}$$

$$LOD = \log_{10} \frac{L(n_{1,+}, n_{1,-}, n_{1,o} \mid H_1)}{L(n_{0,+}, n_{0,-}, n_{0,o} \mid H_0)} = \log_{10} \frac{\theta_{1,+}^{n_+} \theta_{1,-}^{n_-} \theta_{1,o}^{n_o}}{\theta_{0,+}^{n_+} \theta_{0,-}^{n_-} \theta_{0,o}^{n_o}} \tag{6}$$

4 Results

We tested the performance of HMM_CGH model with lymphoid malignancies as an application. The data was provided by the LLMPP and SPECS projects [24, 25]. We used a total of 190 cases of diffuse large B-cell lymphoma (DLBCL) with GEP performed on Affymetrix HG-U133 Set arrays (Santa Clara, CA) and CGH performed by Vysis conventional CGH kits (Downers Grove, IL) which detects gain/loss with the resolution of cytobands. Due to the technical limitation, our conventional CGH experiments could not precisely detect signals for small-sized chromosomes, such as chromosomes 19, 20, 21, 22 and Y. Those five chromosomes were excluded from this study.

We handle chromosomes individually since each chromosome is a well-organized, condensed structure both physically and functionally. The spatial order of the genes on each chromosome was preserved according to the NCBI Human Genome database Build 36.2 [26]. The paired GEP and CGH data are preprocessed separately before integrating them into the model. The software packages BRB-Array Tool [27] were used to analyze the GEP data. We used 1.5-fold change as the threshold to determine over- or under-expression of genes in tumors as compared to "normal" samples. For a large series of tumor cases, the median expression of them can be used as a good approximation of the expression in normal samples. For CGH data, signal ratios greater than 1.25 or less than 0.75 were considered as chromosomal gains or losses, respectively. So GEP and CGH data were translated into symbolic format which can be directly fed into HMM_CGH model—*H*, *L* or *M* for high (> 1.5), low (< 0.5) or medium (between 0.5 and 1.5) expression, and "+", "-" or "*o*" for gain (>1.25) , loss (<0.75) or normal (between 0.75 and 1.25) chromosomal alteration state.

Applying the 190 DLBCL cases to the HMM_CGH model, we evaluate the model in the following three subsections.

4.1 Gain/Loss Pattern Comparison of HMM_CGH, CGH and GEP

We compared chromosomal gain/loss patterns from HMM_CGH prediction, experimental CGH and GEP observations for each chromosome, case by case. Fig. 5 showed an example of the comparison on a few cases randomly selected from the cases which showed chromosomal alterations on chromosome 1 based on CGH. Good concordance of gains/losses was observed between HMM_CGH and CGH, while

GEP observation showed three straight lines, shuffling gain, loss and normal regions. The pattern comparison shown in Fig. 5 is typical of the compositions for all tumor cases and all chromosomes. This suggested that GEP raw observations alone couldn't gave good indications on chromosomal gain/loss, while HMM_CGH could be able to capture more information buried in the GEP data.

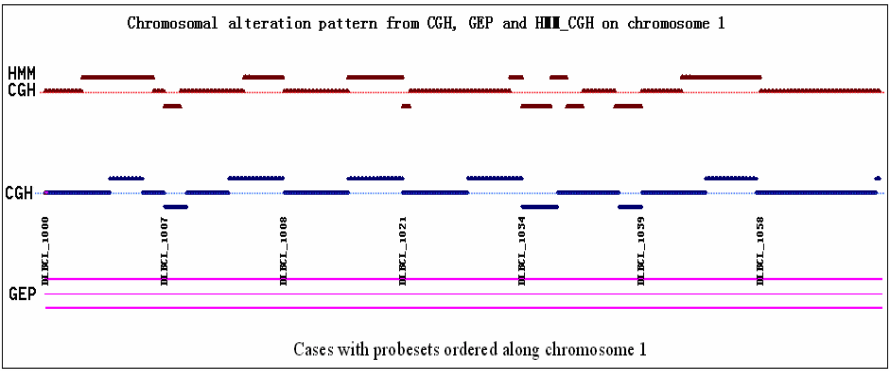


Fig. 5. Comparison of chromosomal gains/losses from HMM_CGH (*red*), CGH (*blue*) and GEP observations (*pink*) of a few randomly selected cases on chromosome 1. It is typical of the comparisons for all cases and chromosomes. For each case, the chromosome is oriented from pter (*left*) to qter (*right*). The dotted lines are the baselines indicating "*o*", and the solid lines above and below indicate "+" and "-" regions, respectively.

4.2 Gain/Loss Prediction Comparison on Probeset of HMM_CGH and GEP

In Fig. 6, we showed how HMM_CGH improved the gain/loss prediction from the GEP raw observations in terms of sensitivity, specificity and accuracy for each chromosome, using the method described in 3.1. Table 1 showed the statistics of sensitivity, specificity and accuracy for all chromosomes. In general, sensitivity was improved from 30% in GEP to 80% in HMM_CGH, specificity from 80% to 90% and accuracy from 80% to 90%, for both gain and loss prediction, and sensitivity from 60% to 85%, specificity from 40% to 80% and accuracy from 60% to 85%, for normal region prediction.

We noticed that the prediction on some chromosomes was not good, such as chromosomes 4 and 17 for gain, and chromosomes 5, 9, 11, 12 and 16 for loss. We looked at the CGH data to check how gains and losses were distribution over tumor cases for each chromosome. We found that some chromosomes have very small number of cases having gain or loss on them— gain: chr 4 (7 cases) and chr 17 (10 cases); loss: chr 5 (1 case), chr 9 (10 cases), chr 11 (1 case), chr 12 (2 cases) and chr 16 (3 cases), out of a total of 190 DLBCL cases. Hence, it is anticipated that without a sufficient number of cases to train some particular chromosomes, the model couldn't predict well on those chromosomes.

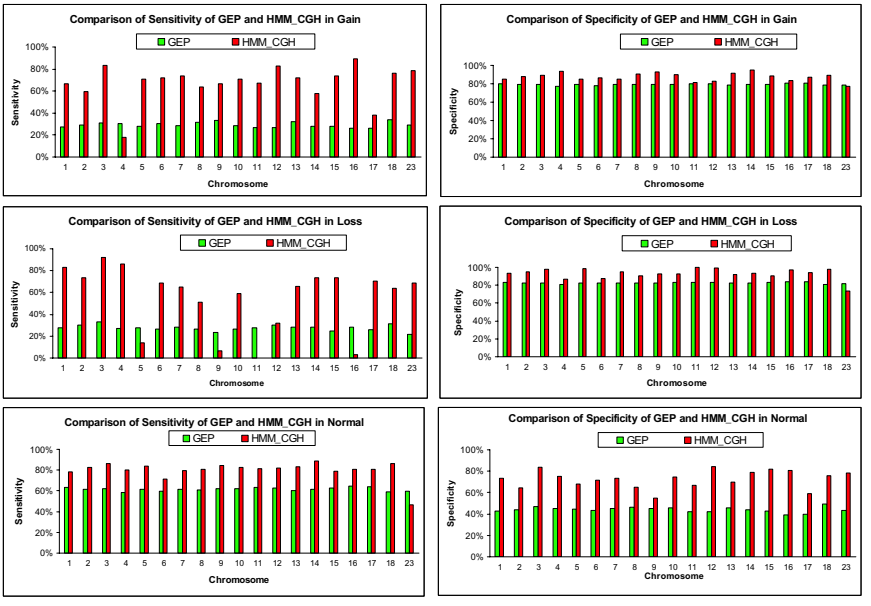


Fig. 6. Sensitivity (*left*) and specificity (*right*) in predicting gain (*upper*), loss (*middle*) and normal (*lower*) regions by HMM_CGH (*red*) and GEP observations (*green*). The accuracy figures are not shown here— they are similar to the specificity figures for gain and loss and similar to the sensitivity figures for normal.

Table 1. Statistics of sensitivity, specificity and accuracy for all chromosomes in HMM_CGH and GEP observations

		Sensitivity	Specificity	Accuracy
Gain	GEP	29.22±2.28%	79.33±0.85%	76.53±1.75%
	HMM_CGH	67.34±16.39%	87.51±4.45%	86.73±4.26%
Loss	GEP	27.50±2.63%	82.55±0.84%	81.41±1.75%
	HMM_CGH	55.26±29.25%	92.96±6.08%	92.51±6.18%
Normal	GEP	61.64±1.74%	44.08±2.39%	60.36±1.93%
	HMM_CGH	79.96±8.86%	72.65±8.15%	79.62±7.72%

4.3 Gain/Loss Comparison on Cytoband of HMM_CGH and CGH

After applying the smoothing algorithm as described in 3.2, we show in Fig. 7 the gain and loss comparison on cytoband from HMM_CGH prediction and actual CGH experiments. A good agreement was observed on most of the chromosomes. Table 2 listed the high-frequency gain/loss regions which are concordant between HMM_CGH and CGH from Fig. 7. In addition, Fisher's Exact was used to determine the nonrandom association of gains/losses between HMM_CGH and CGH. In Fisher's Exact, the sums of the gains and losses for HMM_CGH and CGH are compared on a band-by-band basis and 90% or 80% of the cytobands are similar in the two groups if p-value was set at <1% or <5%, respectively.

From the prediction, we realized that there are some cytobands showing no gains or losses at all for all the cases in HMM_CGH, while a considerable number of cases were observed with gain or loss in those regions in CGH, such as 1q11, 4p11, 4q11, 5p11, 6q11, 9q11, 12q11, 13p, 14p and 15p. Then we looked into the detail of the probeset distribution on HG-U133 chips and found that there were no probesets

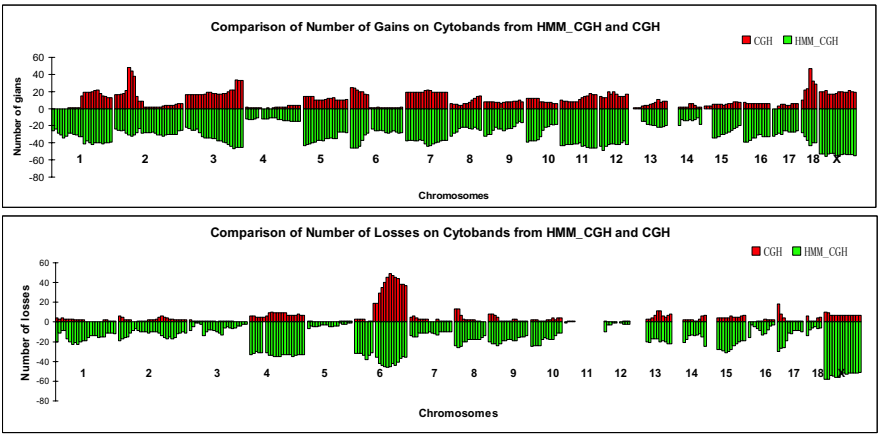


Fig. 7. Gain (*upper*) and loss (*lower*) comparison on the cytobands between HMM_CGH (*green*) and CGH (*red*). Cytobands are ordered from pter to qter for each chromosome.

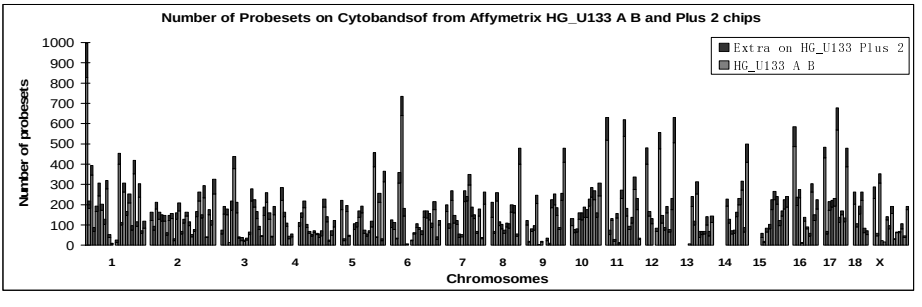


Fig. 8. Distribution of probesets on Affymetrix HG-U133 chips. Cytobands are ordered from pter to qter for each chromosome. The bottom bars stand for the HG-U133 Set and the top bars with the dark color indicate the number of probesets added in HG-U133 Plus 2.

Table 2. Consensus gain/loss regions from HMM_CGH and CGH

	Consensus regions from HMM_CGH and CGH
Gain	1q, 2p16-p14, 3q27-q29, 5p, 6p25-p21, 7q11-q22, 8q23-q24, 10p, 11q, 12q11-q15, 13q31-q34, 18q21, X
Loss	1p, 2p25-p24, 2q21-q24, 4q13-q26, 6q, 7p, 8p23-p21, 9p24-p21, 9q21-q22, 13q, 14q31-q32, 17p13, 18q11, X

selected from those regions in the array design from Affymetrix because no annotated genes were described from public databases for those regions (Fig. 8). Those regions are considered as "gene deserts" and out of scope of our prediction.

5 Conclusions

In this paper, we developed a novel computational model, HMM_CGH, to derive genomic alterations from GEP data in tumors. HMM_CGH was constructed on a hidden Markov model, which was trained in the light of the paired experimental GEP and CGH data on a sufficient number of cases, and then applied to new cases for the prediction of chromosomal gains/losses from their GEP data. The prediction performance of HMM_CGH predictor was tested on 190 cases of diffuse large B-cell lymphoma using cross validation. The results showed that HMM_CGH predictor reached 80% sensitivity, 90% specificity and 90% accuracy in predicting both gains and losses. HMM_CGH can be generally applied to other types of tumors to enhance the detection of genomic alterations.

Acknowledgments. This work was supported by the NIH grant number P20 RR16469 from the INBRE program of the National Center for Research Resources and by U.S. Public Health Service grants CA36727 and CA84967 by the National Cancer Institute, Department of Health and Human Services.

References

1. Golub, T.R., Slonim, D.K., Tamayo, P., Huard, C., Gaasenbeek, M., et al.: Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. Science 286, 531–537 (1999)
2. Ramaswamy, S., Golub, T.R.: DNA microarrays in clinical oncology. J. Clin. Oncol. 20, 1932–1941 (2002)
3. Fridlyand, J., Pinkel, S.A., Albertson, D., Jain, D.G.: AN: Hidden Markov Models Approach to the Analysis of Array CGH Data. J. Multivariate Anal. 90, 132–153 (2004)
4. Olshen, A., Venkatraman, E.: Change-point analysis of array-based comparative genomic hybridization data. In: Proceedings of Joint Statistical Meetings, pp. 2530–2535 (2002)
5. Snijders, A.M., Nowak, N., Segraves, R., Blackwood, S., Brown, N., et al.: Assembly of microarrays for genome-wide measurement of DNA copy number. Nat. Genet. 29, 263–264 (2001)
6. Wang, P., Kim, Y., Pollack, J., Narasimhan, B., Tibshirani, R.: A method for calling gains and losses in array CGH data. Biostatistics 6, 45–58 (2005)
7. Orsetti, B., Nugoli, M., Cervera, N., Lasorsa, L., Chuchana, P., et al.: Genomic and expression profiling of chromosome 17 in breast cancer reveals complex patterns of alterations and novel candidate genes. Cancer Res. 64, 6453–6460 (2004)
8. Clark, J., Edwards, S., John, M., Flohr, P., Gordon, T., et al.: Identification of amplified and expressed genes in breast cancer by comparative hybridization onto microarrays of randomly selected cDNA clones. Genes Chromosomes Cancer 34, 104–114 (2002)
9. Kauraniemi, P., Barlund, M., Monni, O., Kallioniemi, A.: New amplified and highly expressed genes discovered in the ERBB2 amplicon in breast cancer by cDNA microarrays. Cancer Res. 61, 8235–8240 (2001)

10. Monni, O., Barlund, M., Mousses, S., Kononen, J., Sauter, G., et al.: Comprehensive copy number and gene expression profiling of the 17q23 amplicon in human breast cancer. PNAS 98, 5711–5716 (2001)
11. Pollack, J.R., Sorlie, T., Perou, C.M., Rees, C.A., Jeffrey, S.S., Lonning, P.E., Tibshirani, R., Botstein, D., Borresen-Dale, A.L., Brown, P.O.: Microarray analysis reveals a major direct role of DNA copy number alteration in the transcriptional program of human breast tumors. PNAS 99, 12963–12968 (2002)
12. Hyman, E., Kauraniemi, P., Hautaniemi, S., Wolf, M., Mousses, S., et al.: Impact of DNA amplification on gene expression patterns in breast cancer. Cancer Res. 62, 6240–6245 (2002)
13. Phillips, J.L., Hayward, S.W., Wang, Y., Vasselli, J., Pavlovich, C., et al.: The consequences of chromosomal aneuploidy on gene expression profiles in a cell line model for prostate carcinogenesis. Cancer Res. 61, 8143–8149 (2001)
14. Virtaneva, K., Wright, F.A., Tanner, S.M., Yuan, B., Lemon, W.J., et al.: Expression profiling reveals fundamental biological differences in acute myeloid leukemia with isolated trisomy 8 and normal cytogenetics. PNAS 98, 1124–1129 (2001)
15. Varis, A., Wolf, M., Monni, O., Vakkari, M.L., Kokkola, A., et al.: Targets of gene amplification and overexpression at 17q in gastric cancer. Cancer Res. 62, 2625–2629 (2002)
16. Linn, S.C., West, R.B., Pollack, J.R., Zhu, S., Hernandez-Boussard, T., et al.: Gene expression patterns and gene copy number changes in dermatofibrosarcoma protuberans. Am J. Pathol. 163, 2383–2395 (2003)
17. Hughes, T.R., Roberts, C.J., Dai, H., Jones, A.R., Meyer, M.R., et al.: Widespread aneuploidy revealed by DNA microarray expression profiling. Nat. Genet. 25, 333–337 (2000)
18. Bea, S., Zettl, A., Wright, G., Salaverria, I., Jehn, P., et al.: Diffuse large B-cell lymphoma subgroups have distinct genetic profiles that influence tumor biology and improve gene-expression-based survival prediction. Blood 106, 3183–3190 (2005)
19. Salaverria, I., Zettl, A., Bea, S., Moreno, V., Valls, J., et al.: Specific Secondary Genetic Alterations in Mantle Cell Lymphoma Provide Prognostic Information Independent of the Gene Expression-Based Proliferation Signature. J. Clin. Oncol. 25, 1216–1222 (2007)
20. Iqbal, J., DeLeeuw, R.J., Srivastava, G., Geng, H., Patel, K., et al.: High resolution genomic mapping and gene expression analysis of chromosomal aberrations in natural killer malignancies (submitted)
21. Finishing the euchromatic sequence of the human genome. Nature 431, 931–945 (2004)
22. Shi, L., Reid, L.H., Jones, W.D., Shippy, R., Warrington, J.A., et al.: The MicroArray Quality Control (MAQC) project shows inter- and intraplatform reproducibility of gene expression measurements. Nature biotechnology 24, 1151–1161 (2006)
23. Durbin, R., Eddy, S., Krogh, A., Mitchison, G.: Biological sequence analysis: Probabilistic models of proteins and necleic acids. Cambridge Unisersity Press, New York (1998)
24. Lymphoma/Leukemia Molecular Profiling Project (LLMPP), `http://www.cancerdiagnosis.nci.nih.gov/specs/index.htm`
25. Strategic Partnering to Evaluate Cancer Signatures (SPECS), `http://www.cancerdiagnosis.nci.nih.gov/specs/index.htm`
26. NCBI: Homo sapiens (human) genome view (2006), `http://www.ncbi.nlm.nih.gov/projects/mapview/map_search.cgi?taxid=9606`
27. Simon, R., Peng, A.: BRB-Array Tool, `http://linus.nci.nih.gov/BRB-ArrayTools.html`

Evolutionary Algorithm for Feature Subset Selection in Predicting Tumor Outcomes Using Microarray Data

Qihua Tan[1,2], Mads Thomassen[1], Kirsten M. Jochumsen[1], Jing Hua Zhao[3]
Kaare Christensen[2], and Torben A. Kruse[1]

[1] Dept. of Biochemistry, Pharmacology and Genetics, Odense University Hospital, Sdr. Boulevard 29, DK-5000, Odense C, Denmark
qtan@health.sdu.dk
[2] Epidemiology, Institute of Public Health, University of Southern Denmark, J.B. Winsløws Vej 9B, DK-5000, Odense C, Denmark
[3] MRC Epidemiology Unit, Institute of Metabolic Science, Box 285, Addenbrooke's Hospital, Hills Road, Cambridge, CB2 0QQ, UK

Abstract. Feature subset selection for outcome prediction is a critical issue in large scale microarray experiments in cancer research. This paper introduces an integrative approach that combines significant gene expression analysis, the genetic algorithm and machine learning for selecting informative gene markers and for predicting tumor outcomes including survival outcomes. In case of survival data, full use of individual's survival information (both censored and uncensored) is made in selecting informative genes for survival outcome prediction. Applications of our method to published microarray data on epithelial ovarian cancer survival and breast cancer metastasis have identified prognostic genes that predict individual survival and metastatic outcomes with improved power while basing on considerably shorter gene lists.

1 Introduction

Feature subset selection for outcome prediction is a critical issue in large scale microarray experiments in cancer research (Saeys et al. 2007). The development of a powerful prognostic profile requires selecting informative features or markers from a large pool of candidate genes that are present on the arrays. It is well known that a major challenge in microarray analysis is the large number of variable (genes) and the small number of samples which creates the problem of multiple testing (Chen 2007). As a result, simply picking up the significant genes to use as prognostic signatures can result in poor performance of the classifier due to inclusion of false positive genes or significant genes with low impact on classification (Wei and Billings 2007). In addition to the large number of genes, survival analysis of microarray gene expression data is further complicated by issues concerning time-to-event data such as censoring which is a unique feature of survival data. In this case, making efficient use of the observed survival information is crucial in building a good performance prediction model.

In this paper, we introduce an integrative approach that combines significant gene expression analysis, the genetic algorithm and machine learning for selecting

I. Măndoiu, R. Sunderraman, and A. Zelikovsky (Eds.): ISBRA 2008, LNBI 4983, pp. 426–433, 2008.

informative gene markers and for predicting outcomes including survival outcomes. In predicting tumor survival, full use of individual's survival information (both censored and uncensored) is made in selecting informative genes. The method is applied to published datasets from microarray studies on epithelial ovarian cancer survival (Spentzos et al. 2004) and on breast cancer metastasis (van't Veer et al. 2002). Results from our analysis will be compared with that from the original studies.

2 Methods

2.1 Preliminary Gene Filtering

We start our analysis with preliminary gene filtering. Gene filtering is necessary because (1) a prognostic gene should be a significant gene that is associated with the outcome phenotype (Baker and Kramer 2006) and (2) gene filtering can help to remove redundant or uninformative genes from subsequent analysis. To do that, expression for each of the genes is tested for its statistical significance on the outcome under interest using differential gene expression analysis methods such as t-test, ANOVA. For survival data, we apply the univariate Cox regression model to assess the marginal association between the expression of each gene and survival time. Insignificant genes are filtered out using a predefined type I error rate. Note that, in order to avoid "information leak" due to involvement of testing set in gene selection, gene filtering is based on the training set only.

2.2 The Genetic Algorithm

The genetic algorithm (GA) is an adaptive searching method that mimics the natural selection process in evolutionary genetics (Stifanini and Camussi 2000). It is based on a population of competing solutions evolved over time by recombination /cross-over, mutation and selection to converge to an optimal solution for a defined fitness function. Instead of a single solution, multiple solutions are computed and compared to search for the optimum. GA is further featured by its robustness to the size of search space and the underlying multivariate distribution assumption. During evolution, we retain half of the chromosomes that give the highest fitness values without mutation and cross-over and produce the other half of the new chromosomes by recombination and mutation (Li et al. 2005). The retained and the new chromosomes are combined to enter a new generation or iteration. For each feature subset *g*, we set GA to maximize the fitness function defined as the inverse of the following function

$$Z = w[1/c(g)] + (1-w)s(g)$$

Here *s(g)* is the number of selected genes in the feature subset; *c(g)* is the total accuracy calculated from cross-validation; *w* is the weight to be chosen for balancing the validation accuracy and number of genes in the chromosome. By choosing a proper *w*, maximizing the fitness function is equivalent to maximizing the prediction accuracy while limiting the number of genes to be selected.

2.3 Classification Model Building Using SVM

The support vector machines (SVM) is a popular supervised machine learning algorithm widely in use in microarray studies (Brown et al. 2000). SVM builds a hyper plane that separates the training set with maximal discriminative margin. This plane is used to classify new samples in the testing set. Based on the genes selected by GA from the training data, a classification model or classifier is trained using the training data and then applied to the testing data to classify the testing samples. Here, we choose exactly the same samples for training and testing as in the original studies of the published data. This is to enable a fair comparison of our method to the original studies. The whole process from feature selection to model training and testing is illustrated in Figure 1.

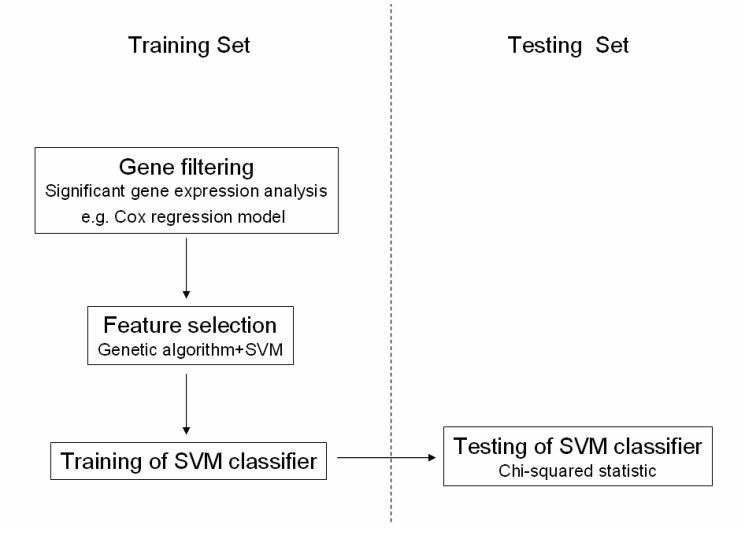


Fig. 1. Flow chart showing the whole data analysis process. Depending on the data, different statistical tests can be applied for filtering the genes.

All calculations are done under the free R programming environment for statistical computing using the free R packages *survival* for Cox regression, *genalg* for GA and *e1071* for SVM.

3 Applications

3.1 Ovarian Cancer Survival Data

We first apply our approach to microarray data on cancer survival from Spentzos *et al.* (2004) who reported prognostic significance of gene expression profiling in survival of epithelial ovarian cancer in a sample of 68 patients using Affymetrix U95A2 array containing approximately 12,000 genes. Their study identified a 115-gene signature that predicted patients with unfavorable and favorable survival outcomes at a significance level of p=0.004. In our analysis, we follow exactly their way of

dividing the samples for training and testing, i.e. 34 samples for training and 34 for testing using exactly the same samples in each group as did in the original study. Moreover, we also adopt the step-wised strategy by Spentzos et al. (2004) for fine tuning in training the model. That is we first train a classifier based on 14 extreme samples (7 shortest survivors without censoring and 7 longest survivors) to classify the remaining training samples in the middle into favorable and unfavorable groups. Then the whole training set together with their group membership is used to train the final model. In feature selection, we set *w* to 0.99 to give very high weight to maximize the cross-validation accuracy calculated using 5 folds cross-validation. Genes are first filtered according to their significance level in affecting individual survival by applying the univariate Cox regression model to the expression data in the training set. We set the criterion for gene filtering to $p<0.03$ (226 genes) for limiting the number of genes in the genetic algorithm while taking into account of multiple testing. Based on the 226 filtered significant genes which constitutes the feature space, our GA feature selection procedure (population size=200; generation size=100; mutation rate=0.2) identified a 5-gene signature that classifies our testing samples into favorable (15 individuals) and unfavorable (19 individuals) groups (Figure 2). The Affymetrix probe IDs and gene names for the 5 prognostic genes are shown in Table 1.

The mean survival time for the unfavorable group is 30 months while that for the favorable is not yet reached in the observation time (Figure 3). Statistical test on differential survival between the two groups has a log rank χ^2=9.834 with 1 degree of freedom and a p-value of 0.0017. Note that this higher statistical significance is achieved by a 5-gene signature instead of 115.

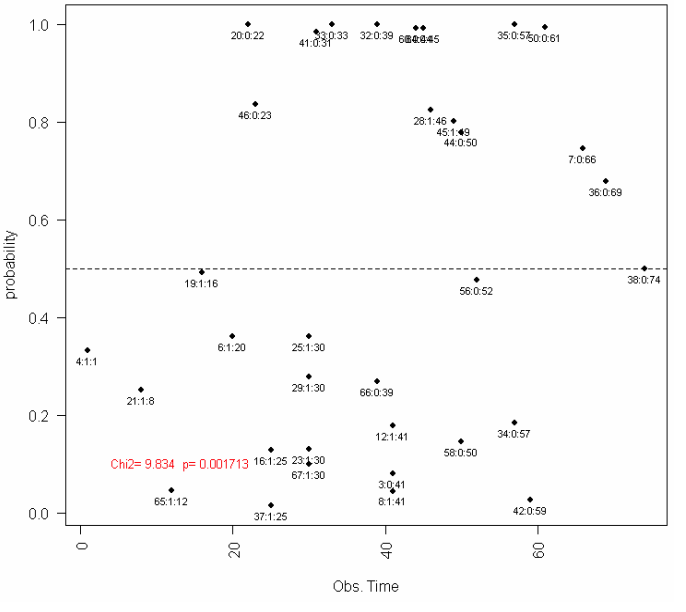


Fig. 2. SVM probability for favorable survival. Each sample is labeled by its ID followed by censoring status and follow-up time. Some censored samples with short observation time are predicted as favorable survivors.

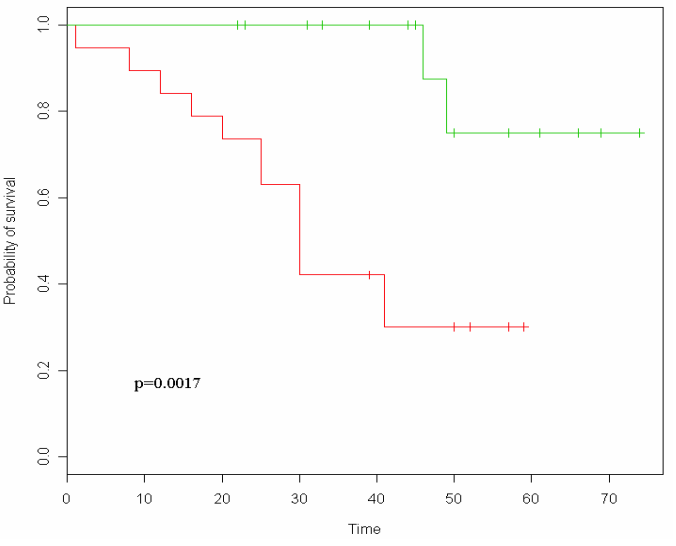


Fig. 3. Kaplan-Meier survival curves for the favorable (upper) and unfavorable (lower) groups. Most of the longest survivors are predicted as favorable.

3.2 Breast Cancer Metastasis Data

Using gene expression profiling, van't Veer et al. (2002) developed a 70-gene signature that predicts breast cancer metastasis within 5 years with high accuracy. The same data was re-analyzed by Thomassen et al. (2006) using similar training (61 samples: 31 metastasis and 30 non-metastasis) and testing (180 samples: 42 metastasis and 138 non-metastasis) sets as in original study but using SVM as classifier. Their analysis produced a sensitivity of 83% and a specificity of 60%. Following our procedure described in Figure 1, we first filtered the 24,496 genes by dropping genes with $p>0.001$ leaving 261 significant genes (feature space) for submitting to GA (population size=1000; generation size=100; mutation rate=0.25). By setting w to 0.999, a 9-gene signature was developed by GA using the 61 training samples. A final classifier was trained using the 9-gene signature and 61 training sample. This classifier predicts metastatic outcomes of the 180 testing samples with a sensitivity of 60% and specificity of 74% when the cut-off for SVM probability is set to 0.5. Table 1 has the probe IDs for the 9-gene signature from the raw data together with their gene names. Figure 4 displays the SVM probability for the 180 testing samples from which a clear trend of separation of metastasis and non-metastasis can be seen.

Based on the 9-gene profile and using a cut-off for SVM probability of 0.5, the result from Figure 4 has a higher specificity (74%) but a lower sensitivity (60%) as compared to Thomassen et al. (2006) (sensitivity 83%, specificity 60%) using a much larger 70-gene signature. However, the total accuracies are all 71% in both analyses. Similar to van't Veer et al. (2002), one can easily move the cut-off downward to achieve a high sensitivity above 80% while still having most of the non-metastasis samples correctly classified.

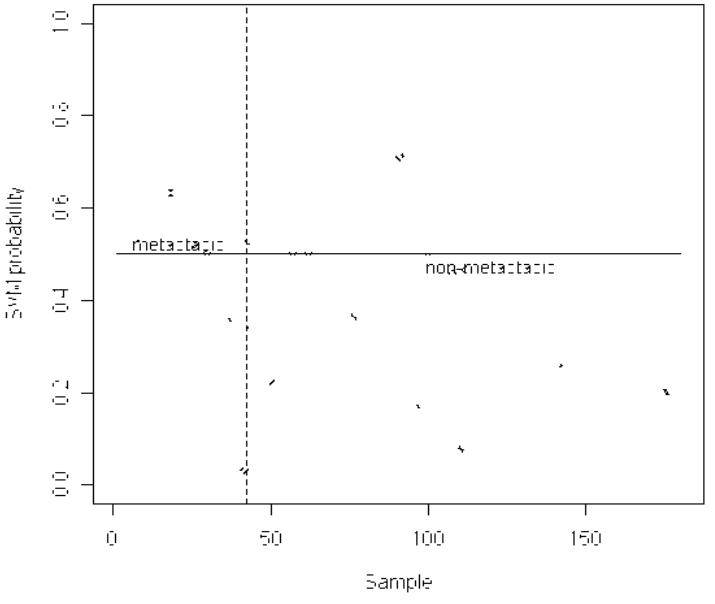


Fig. 4. SVM probability for the 180 testing samples. A clear separation of metastasis and non-metastasis patients can be seen.

Table 1. Probe IDs and gene names for the selected prognostic features

Probe ID	Gene name
Ovarian cancer	
34955_at	ABCC4
40307_at	ATAD2B
38750_at	NOTCH3
413_at	HOXD9
285_g_at	HIST1H2AG
Breast cancer	
Contig29682_RC	CDK3
NM_003293	TPS1
NM_003403	YY1
Contig23188_RC	MS4A7
AF161414	MED11
Contig32739	ZNF117
NM_017855	FLJ20513
NM_001446	FABP7
Contig32087_RC	AK125443*

*A cluster of ESTs.

4 Discussion

Through example applications, we have shown that the evolutionary method can be used for feature subset selection for prognostic analysis of tumor outcome using microarray gene expression data. Very informative subset of genes can be identified by GA when applied to significant genes filtered by applying conventional statistical models for differential gene expression analysis. Application of our method to published data on epithelial ovarian cancer has captured a subset of statistically significant genes (5 genes) that discriminates cancer patients with unfavorable and favorable survival outcomes (p=0.002) which outperforms the original result (p=0.004) from a 115-gene signature developed by Spentzos et al. (2004). It is interesting to compare our Figure 3 with the result from the original analysis in Figure 3A by Spentzos et al. (2004). In both figures, the mean survival for the unfavorable group is 30 months. However, one censored sample with short observation time originally clustered as unfavorable is now predicted as favorable. Two samples with long observation time grouped as unfavorable are now predicted as long survivors. Our prediction is more meaningful as the longest survivor is put into the favorable group by our short signature genes while it was put in the unfavorable group by the original analysis in Spentzos et al. (2004). Note that the use of Cox regression model on gene expression data makes full use of individual survival information (both censored and uncensored) in the process of gene filtering.

A good classification signature should be a minimal subset of genes that is not only differentially expressed but also contains most relevant genes without redundancy (Baker and Kramer 2006). Filtering the significant genes using statistical tests for differential gene expression analysis on the training samples can largely reduce the searching space for the genetic algorithm to converge the optimal solution. Our experience with GA showed that, without the weighting scheme in the fitness function, the larger the number of genes submitted the more genes can be selected by GA. With the weighting scheme, the number of selected genes can be controlled when a proper weight is assigned. Empirical applications indicate that our strategy enables good prediction for the testing samples by a subset of genes that contains much lower number of genes as compared with the original study.

It is interesting that none of our selected genes overlap with the gene lists selected from the original studies. Ein-Dor *et al*.(2005) reported that the set of out-come predictive genes is not unique due to the existence of multiple genes that are correlated with the outcomes and some of them may have only small differences in their correlations. Results from our study showed that GA can be a useful tool for finding the subset of predictive genes that are both informative and most representative.

Acknowledgements

We thank Dr. Dimitrios Spentzos at Beth Israel Deaconess Medical Center in Boston for help in accessing their data and for providing the list of their signature genes. This work was partially supported by the US National Institute on Ageing (NIA) research grant NIA-P01-AG08761.

References

Baker, S.G., Kramer, B.S.: Identifying genes that contribute most to good classification in microarrays. BMC Bioinformatics 7, 407 (2006)

Brown, M.P., Grundy, W.N., Lin, D., Cristianini, N., Sugnet, C.W., Furey, T.S., Ares, M., Haussler, D.: Knowledge-based analysis of microarray gene expression data by using support vector machines. Proc. Natl. Acad. Sci. USA. 97, 262–267 (2000)

Chen, J.J.: Key aspects of analyzing microarray gene-expression data. Pharmacogenomics 8, 473–482 (2007)

Ein-Dor, L., Kela, I., Getz, G., Givol, D., Domany, E.: Outcome signature genes in breast cancer: is there a unique set? Bioinformatics 21, 171–178 (2005)

Li, L., Jiang, W., Li, X., Moser, K.L., Guo, Z., Du, L., Wang, Q., Topol, E.J., Wang, Q., Rao, S.: A robust hybrid between genetic algorithm and support vector machine for extracting an optimal feature gene subset. Genomics 85, 16–23 (2005)

Saeys, Y., Inza, I., Larrañaga, P.: A review of feature selection techniques in bioinformatics. Bioinformatics 23, 2507–2517 (2007)

Spentzos, D., Levine, D.A., Ramoni, M.F., Joseph, M., Gu, X., Boyd, J., Libermann, T.A., Cannistra, S.A.: Gene expression signature with independent prognostic significance in epithelial ovarian cancer. J. Clin. Oncol. 22, 4700–4710 (2004)

Stefanini, F.M., Camussi, A.: The reduction of large molecular profiles to informatic components using a genetic algorithm. Bioinformatics 16, 923–931 (2000)

Thomassen, M., Tan, Q., Eiriksdottir, F., Bak, M., Cold, S., Kruse, T.A.: Prediction of metastasis from low-malignant breast cancer by gene expression profiling. International Journal of Cancer 120, 1070–1075 (2006)

van t̀ Veer, L.J., Dai, H., van de Vijver, M.J., et al.: Gene expression profiling predicts clinical outcome of breast cancer. Nature 415, 530–536 (2002)

Wei, H., Billings, S.A.: Feature subset selection and ranking for data dimensionality reduction. IEEE Transactions on Pattern Analysis and Machine Intelligence 29, 162–166 (2007)

Incorporating Knowledge of Topology Improves Reconstruction of Interaction Networks from Microarray Data

Peter Larsen[1], Eyad Almasri[2], Guanrao Chen[3], and Yang Dai[2]

[1] Core Genomics Laboratory, Research Resource Center (MC937), University of Illinois at Chicago, 835 South Wolcott Avenue, Chicago, IL 60612, USA
[2] Department of Bioengineering (MC063), University of Illinois at Chicago, 851 South Morgan Street, Chicago, IL 60607, USA
[3] Department of Computer Science (MC152), University of Illinois at Chicago, 851 South Morgan Street, Chicago, IL 60607, USA
{plarsen, ealmas1, gchen4, yangdai}@uic.edu

Abstract. Reconstruction of biological interaction networks from high-throughput experimental data is one of the most challenging problems in bioinformatics. These networks have specific topologies, whose characteristics are defined by evolutionary relationships between proteins and the physical limitations imposed on proteins interacting in three-dimensional space. In this study, a method is proposed applying the topology of known biological networks to the analysis of microarray data for protein-protein binding interactions. In this method, genomic biological networks are derived from the body of published scientific literature. The numbers of interacting neighbors for proteins of specific molecular functions are observed. That information is used in the analysis of microarray expression data to regenerate biological networks using a rank-based algorithm, Gene Ontology Restricted Value Neighborhood (GRV-N). The results of this analysis demonstrate that incorporating knowledge of network topology improves the ability of expression analysis to reconstruct interaction networks with a high degree of biological relevance.

Keywords: Rank-based algorithm, Gene Ontology, Gene expression, Co-expression network, Network topology.

1 Introduction

The reconstruction of genetic networks based on microarray gene expression data is one of the most challenging tasks in bioinformatics. The type of interactions considered in this study is protein-protein binding interactions and it is assumed that proteins that interact are also likely to be co-expressed as observed by microarray expression analysis. The typical approach to co-expression analysis is through the computation of correlation coefficient between a pair of gene expression profiles. The networks reconstructed based on these methods are called reference networks [1-3].

I. Măndoiu, R. Sunderraman, and A. Zelikovsky (Eds.): ISBRA 2008, LNBI 4983, pp. 434–443, 2008.

As pointed out in [4-6], large scale networks, such as the Internet and the scientific collaboration network, show the scale-free property, i.e., the connections or edges in the networks follow the power law distribution. The analyses of many biological networks, including transcription regulatory networks and protein interaction networks revealed that these networks are not random and resemble some scale-free property. Although it is still controversial whether these networks are scale-free, the existence of highly connected nodes in the biological networks is evident.

So far there is little research that has explicitly explored this important property to facilitate the learning of genetic networks from gene expression data. One recent study imposed the scale-free constraint on structure in network inference based on the S-system model [5]. They investigated the performance with a simulated small-scale time-course data. On the other hand, different mechanisms have been employed to explain the formation of the scale-free property in large-scale networks other than biological networks. Most of the suggested models relate to Preferential Attachment [3]. In contrast to modeling network growing, a model with fixed number of nodes and links was proposed recently [5]. By applying local rewiring moves, the network can reach equilibrium states that have the power law degree distribution. Different mechanisms were also proposed to explain specific properties of different types of networks, such as genetic regulatory networks and the World Wide Web [6].

In our previous study [7], we proposed a rank-based network reconstruction algorithm that takes into account the scale-free network topology. The algorithm, named Asymmetric-N, is based on a modification of the Symmetric-N algorithm [8], in which no distinctions are made for nodes in the network. In Asymmetric-N, a network was considered to consist of two types of nodes: Core and Peripheral. The expected neighborhood size of a Core node is far larger than that of a Peripheral node. Use of this distinction permitted Asymmetric-N to recover networks that were not only scale-free but, when applied to the analysis of microarray data, yield networks of greater biological significance compared to the algorithm Symmetric-N. However, Asymmetric-N requires the specification of Core and Peripheral nodes and pre-subscribed neighborhood sizes. The first requirement is relatively easy to be met in the context of reconstruction of transcription regulatory networks as the transcription regulators can be considered as Core nodes and the rest as peripheral nodes. However, the determination of neighborhood sizes is challenging.

The method proposed in this work removes the above requirements through the analysis of functional aspects of known biological networks. Each protein is annotated to several of the 23 yeast GO-Slim Molecular Function (MF) annotations as provided by the *Saccharomyces* Genome Database (SGD) [9]. A large body of known biological interactions has been collected from the databases of Ariadne Genomic Inc.'s 'PathwayStudio' software tool [10]. The neighborhood size of each protein will be determined by the analysis of known interactions in the database of known interactions. With the specified neighborhood sizes of proteins, the GRV-N algorithm is applied.

For the purpose of this study, we restrict protein interactions to those considered to be binding interactions. These are direct, physical relationships in which the molecular nature of the proteins involved should be informative.

We demonstrate that the reconstructed network based on this approach has greater biological relevance compared to methods that do not use any prior biological knowledge. A biologically relevant interaction network is considered to be the one in which a significant proportion of identified interaction occur between protein of the same biological process or in the same sub-cellular location. Such an interaction network is also likely to contain many previously observed interactions. For this study, the SGD GO-Slim Biological Process and Cellular Compartment annotations are considered.

2 Method

The method proposed here requires a database of known protein-protein interactions, an ontology of specific annotations that can be applied to the proteins in these interactions, and a microarray dataset on which the proposed method can be tested.

2.1 Database of Published Protein Binding Interactions

'PathwayStudio' [10] is a bioinformatics tool that identifies possible interactions between gene products through a natural language search algorithm of all available PubMed published abstracts. Given an input set of query genes or gene products, 'PathwayStudio' searches the database of published abstracts, seeking instances in which genes are identified as interacting according to the information found in available PubMed abstracts. The nature of interactions ('expression', 'regulation', 'genetic interaction', 'binding', 'protein modification', and 'chemical modification' as defined in that software package) can be used to screen for specific types of interactions. The software returns the set of interactions with the PubMed references from which those interactions were identified. For this study, the entire *Sachcaromyces* genome was submitted and interactions of type 'binding' were collected for all proteins.

2.2 Gene Ontology

To impose biological knowledge on the set of gene products analyzed, annotation descriptions from the Gene Ontology (GO) [11] were used. There are three ontologies: Molecular Function, Biological Process, and Cellular Component. Molecular Function (MF) annotation describes what gene product does at the molecular level, without specifying where or when the activity takes place in the broader context. Biological Process refers to a biological objective to which a gene product contributes, though GO Biological Process (BP) annotations are not the equivalent of a biological pathway. Cellular Component (CC) annotation refers to the place in the cell where a gene product is found. GO annotations, at their finest level do not describe specific gene products and a given gene product may have multiple GO annotations from each ontology.

The specific GO ontologies considered in this study are the GO-Slim annotations as provided by the *Saccharomyces* Genome Database (SGD) [9], an expert currated selection of high-level annotations from the Biological Process, Molecular Function, and Cellular Component ontologies.

2.3 Microarray Dataset

In this study, a subset of microarray data of cell cycle regulated genes in the budding yeast *Saccharomyces cerevisiae* microarray experiments [12] were used for the validation of the algorithms. These microarray experiments were designed to create a comprehensive list of yeast genes whose transcription levels were expressed periodically within the cell cycle. The gene expressions of cell cycle synchronized yeast cultures were collected over 18 time points taken in 7-minute intervals. This time series covers more than two complete cycles of the cell division. The subset used here is comprised of 998 of the most cyclically regulated genes in the microarray experiments as identified by Cyclic Correlation Coefficients (CCC) [13].

2.4 Gene Ontology Restricted Value Neighborhood Method (GRV-N)

GRV-N is a modification of the algorithm Asymmetric-N [7]. One of the issues with Asymmetric-N algorithm is the need for the specification of neighborhood sizes for core and peripheral nodes. In their work, the determination of these values was empirical. In the present study, we propose a method of assigning neighborhood size for each node in the network based on the information derived from the existing interactions

From the database of all yeast proteins from 'PathwayStudio', 14,345 binding interactions were obtained between 1951 proteins. Each protein is then annotated by its GO-Slim MF annotations. For each annotation, the average number and standard deviation of interacting proteins with this annotation in the 14,345 interactions are calculated (Table 1). The average of all neighborhood sizes is 18.

An individual protein's neighborhood size is then determined by considering its GO-Slim MF annotations and finding its neighborhood size from Table 1. More specifically, the neighborhood size of the protein is determined to be equal to the average neighborhood size for its GO-MF annotation plus k times the standard deviation. Here k is a parameter of integer. If a protein has multiple GO-Slim MF annotations, then the largest possible neighborhood size for that protein is used.

The algorithm for GRV-N follows. Let *NumNodes* represent the total number of nodes in the network; *N* the vector of size *NumNodes* with each entry representing neighborhood size for node *i*; and *CorrelationMatrix* the pre-computed values of the correlation coefficients of gene expression profiles for all pairs of nodes. *PCCThresh* is the threshold below which potential interactions will not be considered. The function *mySort*() returns the other nodes in the sorted order in terms of their 'closeness' or correlation with the selected node.

Algorithm GRV-N

ConstructedNet = *GRV-N*(*NumNodes, CorrelationMatrix, PCCthresh*)
  Step 1: for *i* = 1 to *NumNodes*
    SortedNeighbor[*i*, 1:*NumNodes* - 1] = *mySort*(*i, CorrelationMatrix*);
  Step 2: for *i* = 2 to *NumNodes*
    for *j* = 1 to *i* – 1
      if (*j* is in *SortedNeighbor*[*i*, 1:N_i] and *i* is in *SortedNeighbor*[*j*, 1:N_j] and
        CorrelationMatrix[i, j] > *PCCthresh*)
      then *ConstructedNet*[*i, j*] = *ConstructedNet*[*j, i*] = 1;
      otherwise *ConstructedNet*[*i, j*] = *ConstructedNet*[*j, i*] = 0*;*

Table 1. The average and standard deviations for number of neighbors of proteins grouped by GO-Slim Molecular Function annotation was derived from a database of published interactions from the Ariadne Genomics, 'PathayStudio' database. 14,345 Binding interactions were obtained between 1951 proteins.

GO Molecular Function Annotation	Ave	SD	GO Molecular Function Annotation	Ave	SD
DNA binding	18.1	19.4	Phosphoprotein phosphatase	12.5	13.5
Enzyme regulator activity	15.9	19.1	Protein binding	16.1	19.4
Helicase activity	23.2	28.9	Protein kinase activity	11.8	12.2
Hydrolase activity	16.2	19.1	RNA binding	33.6	26.1
Isomerase activity	28.3	30.5	Signal transducer activity	10.8	13.5
Ligase activity	21.9	21.4	Structural molecule activity	20.6	17.7
Lyase activity	14.7	17.0	Transcription regulator	13.3	16.1
Molecular function unknown	10.4	16.5	Transferase activity	13.8	18.6
Motor activity	12.9	20.2	Translation regulator activity	32.9	26.3
Nucleotidyltransferase activity	23.8	24.5	Transporter activity	9.7	14.0
Oxidoreductase activity	13.5	19.6	Other	10.0	15.7
Peptidase activity	21.4	22.9			

3 Results

To demonstrate the utility of the proposed GRV-N algorithm for construction biologically relevant protein interaction networks, three methods for making interaction networks were compared. The first is GRV-N using a PCC threshold of 0.9. To test that any improvement in network quality is solely due to a restriction on neighborhood size and not the GO annotation-specific neighborhood size, interaction network using a fixed neighborhood size of 18 (N-18) was tested. The method is denoted as Fixed-N(18). PCC alone at thresholds of 0.9, 0.9085, and 0.95 were also tested. The threshold of 0.9085 was selected because it produces an interaction network of nearly the same size as GRV-N and was used to determine if any observed improvement by GRV-N is not solely due to the size of the network. The proportions of identified interactions that share a GO-Slim BP or CC annotations determine the quality of calculated interaction networks.

3.1 Compare GRV-N to PCC and Fixed-N(18)

Results of different methods for determining interaction networks from microarray expression data (Table 2) indicate that there are substantial differences between the methods.

GRV-N has the highest percent of interactions that share a GO-CC annatation, followed by Fixed-N(18) PCC (0.95) has the worst percent interactions that share a GO-CC annotation. Though it is a very small set of interactions, PCC (0.95) has the highest percent interactions that share a GO-BP annotation, followed by GRV-N. All

methods show some improvement over PCC (0.9). It is important to note that all other identified networks are actually a subset of interactions identified by PCC (0.9).

To best understand these results, it is useful to assign a statistical relevance to the observations. Hypergeometric distribution was used to calculate the probability that the number of edges in an interaction network that share a GO annotation out of the total number of interactions identified from the set of interactions in PCC (0.9) could occur by random chance. By that criterion, GRV-N, with pVal-GO-BP=4.85E-5 and pVal-GO-CC=5.44E-08, is far more significantly enriched for interactions sharing a GO annotation than expected by chance than any other method tested. Even for the small network identified by high stringency PCC (0.95), though it had the highest percent of shared GO-BP annotations, is not as statistically significant (pVal-GO-BP=1.97-E2 and pVal-GO-CC=8.67-E2) as the larger GRV-N identified interaction network.

Table 2. Results from several methods for identifying protein-protein interaction networks from cyclically expressed, cell cycle microarray data are summarized here. The methods used are "GRV-N" for GRV-N using a neighborhood size equal to the average of neighbors by GO-Slim MF annotation, "GRV-N(+1SD)" for GRV-N using a neighborhood size equal to the average of neighbors plus one standard deviation by GO-Slim MF annotation, "Fix-N(18)" uses a rank-based method and a maximum neighborhood size of 18, and "PCC (#)" uses Pearson Correlation Coefficient alone and a threshold of (#). "pVal GO-BP" and "pVal-GO-CC" are probabilities of finding the number of interactions that share a GO annotation by selecting from PCC (0.9) interactions at random, as calculated by hypergeometric distribution.

Method	# Interactions	%Same GO-BP	%Same GO-CC	pVal GO-BP	pVal GO-CC
PCC (0.9)	**695**	**26.3**	**30.6**		
GRV-N	539	29.7	35.4	4.85E-05	5.44E-08
GRV-N (+ 1 SD)	684	26.8	31.1	3.37E-02	1.72E-02
Fix-N(18)	608	26.8	32.1	8.02E-02	9.36E-03
PCC (0.9085)	538	27.3	31.0	4.58E-02	7.25E-02
PCC (0.95)	119	32.8	30.3	1.97E-02	8.67E-02

3.2 Effect of Neighborhood Sizes

To determine if the improvement seen in GRV-N method is due to the relevance of the neighborhood sizes used, GRV-N was compared to one thousand networks using randomly selected neighborhood sizes. For the randomly selected neighborhood sizes, each GO-MF annotation was assigned a random value between the minimum and maximum neighborhood sizes from Table 1, ten to thirty four. Significance of GRV-N results was considered to be the frequency at which a network generated with random neighborhood sizes had a better percent of GO-BP or GO-CC shared annotations in the final network.

The average number of interactions in the 1000 random neighborhood networks was 1,251.9 with a standard deviation of 76.1. The frequency at which higher percent

of shared GO annotations were observed in the random neighborhood networks was 0.046 and 0.016 for GO-BP and GO-CC respectively. This indicates that specific neighborhood sizes calculated have a significant effect on deriving a network with biological relevance.

3.3 Shared GO-BP and GO-CC Annotations

Though GRV-N and the specific neighborhood sizes used in calculations have been determined to be statistically significant improvement over other methods tested by considering proportion of identified interactions that share a GO annotation, the final test of an interaction network is its biological utility. Though GRV-N is best at finding interactions that share GO annotation, Table 3 lists the specific GO annotations that are shared and present at greater than 2.5% of total shared interactions in at least one method's network. Though overall differences are slight between methods, the overall set of GO annotations that are shared between interacting proteins are very logical for an experiment investigation cell cycle; the most frequently occurring shared GO annotations are "DNA metabolism", "Cell cycle", and "Response to stress". GRV-N has slightly higher proportions of "Response to stress", "Translation", "RNA metabolic process", and "Protein modification process" annotations.

Table 3. For those protein interactions that share a GO-Slim Biological Process or GO-Slim Cellular Component annotation and are present in at least 2.5% in at least one method's results, the distribution of GO annotation shared are presented here. Four specific networks are considered. "GRV-N" is for GRV-N using a neighborhood size equal to the average of neighbors by GO-Slim MF annotation at a PCC threshold of 0.9. "Fixed-N(18)" uses a maximum neighborhood size of 18 at a PCC threshold of 0.9. PCC(0.9) and PCC(0.95) are for PCC alone at PCC threshold of 0.9 and 0.95 respectively. "GRV-N" Values are highlighted when they are the highest for an annotation.

% GO-Slim Annotation	PCC (0.9)	GRV-N	N-18	PCC (0.95)
Cell wall organization and biogenesis	1.1	1.3	1.2	2.6
Protein modification process	2.7	**3.1**	3.1	2.6
RNA metabolic process	2.7	**3.1**	3.1	0.0
Translation	4.9	**5.6**	5.5	2.6
Organelle organization and biogenesis	7.7	7.5	8.0	2.6
Response to stress	9.3	**10.6**	10.4	7.7
Cell cycle	30.6	26.9	28.2	20.5
DNA metabolic process	36.6	36.9	35.6	61.5
Cellular bud	1.9	2.1	2.1	2.8
Cell wall	1.9	2.1	2.1	2.8
Ribosome	4.2	4.7	4.6	2.8
Cytoplasm	18.3	19.4	19.5	5.6
Chromosome	27.7	28.8	27.7	75.0
Nucleus	40.8	37.2	38.5	11.1

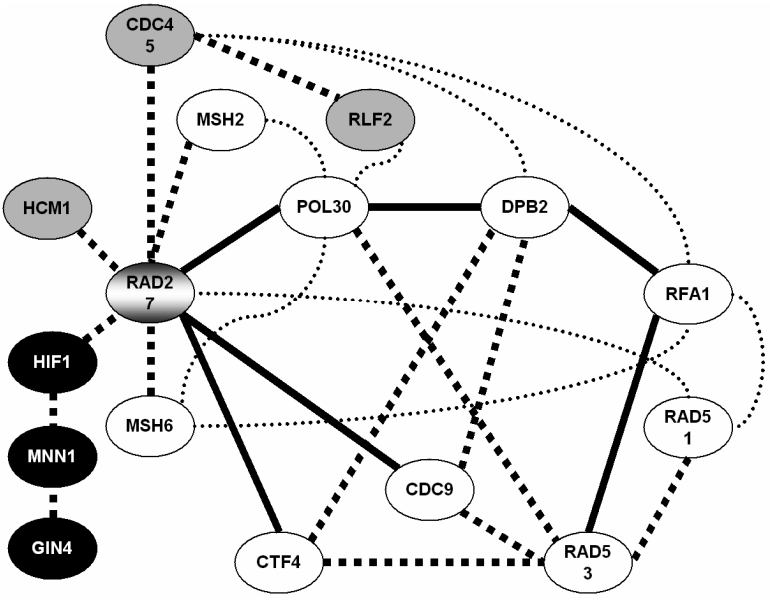


Fig. 1. The largest connected set of proteins identified by GRV-N from analysis of microarray data with interactions that share GO-BP annotations "Response to stress" (white), "RNA metabolic process" (gray), and "Protein modification process" (black). Protein 'RAD27' is annotated with all three GO-BP annotations. Solid lines are interactions identified by GRV-N and previously published, dashed lines are identified by GRV-N but not previously published, and curved, dotted lines are published interactions that were not identified by GRV-N.

The protein interactions in these four enriched specific GO-BP annotations were selected from the set of interactions identified by GRV-N. The largest interconnected set of proteins from these interactions is pictured in Figure 1. This sub-network combines genes involved in DNA replication, base excision repair, and maintenance of genome stability, linked together through the multifunctional nuclease RAD27 [14]. In this figure, 33% of identified interactions correspond to previously published interactions as identified in the database of 'PathwayStudio', suggesting a high degree of biological relevance. The identified set of interacting proteins is further connected by known interactions that were not specifically uncovered by this analysis. This suggests that the identified interactions are indeed portions of larger protein complex. There are a number of reasons that this method might fail to identify previously published interactions. The interactions may be true, but not present in the specific system of the microarray study. They may be true and present in the microarray study, but do not meet the expectation that interacting proteins are necessarily co-expressed. GRV-N identified interactions that are not previously identified in the published literature might also represent useful, biological observations. Although some might be false positives, others might be potentially novel interactions, true and known interactions that are not well represented in 'PathwayStudio', or true protein interactions via some larger protein complex whose elements are not all present in the 998 cyclically expressed genes used in this analysis.

4 Conclusions

In this study, a method, Gene Ontology Restricted Value Neighborhood, GRV-N, was proposed to incorporate knowledge of biological network topology into the analysis of microarray data to construct protein-protein interaction networks.

The results of this study indicate that GRV-N performs better than PCC alone or Fixed-N in its ability to recover interaction networks with interaction pairs that share GO-Slim annotations. The improvements in interaction networks were statistically significant and independent of variables such as network size. The GO-MF annotation-specific neighborhood sizes were significant and informative as demonstrated by a comparison to one thousand networks generated using randomized neighborhood sizes. The interactions identified and the specific GO-Slim ontology annotations shared suggest significant biological relevance of the GRV-N identified networks.

The method proposed here uses a database of protein-protein binding interactions derived from published data and GO-Slim Molecular Function annotations to describe network topology. Certainly, identifying the characteristic topology of additional types of interactions, derived from any of the numerous alternative databases, may provide additional insight into biological interaction networks and utility for analyzing microarray data using GRV-N. Although GO-Slim Molecular Function annotation was used here, there are many possible ways to select ontology annotation to describe proteins that might prove even better at describing protein-protein binding or any additional type of interactions that could be considered such as phosphorylation, methylation, or enzyme regulation.

References

1. Stuart, J.M., et al.: A Gene-Coexpression Network for Global Discovery of Conserved Genetic Modules. Science 302(5643), 249–255 (2003)
2. Zhang, B., Horvath, S., General Framework, A.: for Weighted Gene Co-Expression Network Analysis. Statistical Applications in Genetics and Molecular Biology 4(1), 17 (2005)
3. Ghazalpour, A., et al.: Integrating Genetic and Network Analysis to Characterize Genes Related to Mouse Weight. PLoS Genetics 2(8), 130 (2006)
4. Barabasi, A.L., Bonabeau, E.: Scale-free Networks. Sci. Am. 288, 60 (2003)
5. Farkas, I., et al.: The Topology of the Transcription Regulatory Network in the Yeast, *Saccharomyces cerevisiae*. Physica A-Statistical Mechanics And Its Applications 318(3–4), 601–612 (2003)
6. Albert, R., Barabási, A.L.: Statistical Mechanics of Complex Networks. Reviews of Modern Physics 74(1), 47 (2002)
7. Chen, G., et al.: Rank-Based Edge Reconstruction for Scale-Free Genetic Regulatory Networks. BMC Bioinformatics 9, 75 (2008)
8. Agrawal, H.: Extreme Self-Organization in Networks Constructed from Gene Expression Data. Physical Review Letters 89(26), 268702 (2002)
9. GO Slim Mapper, http://db.yeastgenome.org/cgi-bin/GO/goTermMapper
10. Nikitin, A., et al.: Pathway Studio–the Analysis and Navigation of Molecular Networks. Bioinformatics 19(16), 2155–2157 (2003)

11. Gene Ontology Consortium, The Gene Ontology (GO) Project in 2006. Nucl. Acids Res. 34(suppl. 1), 322–326 (2006)
12. Spellman, P.T., et al.: Comprehensive Identification of Cell Cycle-Regulated Genes of the Yeast *Saccharomyces cerevisiae* by Microarray Hybridization. Mol. Biol. Cell 9(12), 3273–3297 (1998)
13. Larsen, P., et al.: A Statistical Method to Incorporate Biological Knowledge for Generating Testable Novel Gene Regulatory Interactions from Microarray Experiments. BMC Bioinformatics 8, 317 (2007)
14. Liu, Y., et al.: Flap Endonuclease 1: A Central Component of DNA Metabolism. Annu. Rev. Biochem. 73, 589–615 (2004)

Invited Keynote Talk: Data Mining and Statistical Methods for Analyzing Microarray Experiments

Shin-Lian Lo[1], Kwok-Leung Tsui[1], and Benjamin Barwick[2]

[1] School of Industrial and Systems Engineering,
Georgia Institute of Technology, Atlanta, 30332
{slo6,ktsui}@isye.gatech.edu
[2] Department of Human Genetics, School of Medicine,
Emory University, Atlanta, 30033
bbarwick@genetics.emory.edu

Abstract. Deoxyribonucleic acid (DNA) microarrays are part of a promising class of biotechnologies that allow the simultaneous monitoring of expression levels in cells for thousands of genes. One of important issues in microarray experiments is the classification of biological samples and predicting clinical or other outcomes using gene expression data. A closely related issue is the identification of marker genes that have good predictive power for an outcome of interest. Although classification is not a new subject in the statistical literature, the large number of genes with relatively small sample size generated by microarray experiments raises new computational challenges. In this study, the gene expressions of breast cancer tumors are investigated and the performance of several popular classification methods, including decision tree, logistic regression, linear discriminant analysis, and k-nearest neighbor are compared. The results show that certain genes are significantly differentially expressed across groups of patients, and k-nearest neighbor method achieves better performance in class prediction than the other classification methods.

In addition to reviewing and illustrating the implementation of standard statistical tests and classification methods in modeling genome data, we will also address some important issues in the study, such as the role of experimental design (e.g., split-plot experimental design and analysis), the impact of correlation (within plate, between plates, between probe, etc.), the sampling issue in cross validation and training-testing splitting. While these issues have been discussed in simple statistical problems, they have not been well understood by bioinformatics researchers in modeling complex microarray data. In this talk, we will address these issues and their impact on various standard testing and classification methods and illustrate the potential problems through the cancer tumor microarray experiments.

I. Măndoiu, R. Sunderraman, and A. Zelikovsky (Eds.): ISBRA 2008, LNBI 4983, p. 444, 2008.

Seven Variations of an Alignment Workflow - An Illustration of Agile Process Design and Management in Bio-jETI*

Anna-Lena Lamprecht[1,2], Tiziana Margaria[3], and Bernhard Steffen[1]

[1] Chair of Programming Systems, Dortmund University of Technology, 44227 Dortmund, Germany
anna-lena.lamprecht@cs.uni-dortmund.de, steffen@cs.uni-dortmund.de
[2] Center of Applied Proteomics, 44227 Dortmund, Germany
[3] Chair of Service and Software Engineering, Potsdam University, 14482 Potsdam, Germany
margaria@cs.uni-potsdam.de

Abstract. This paper shows how the agility provided by the Bio-jETI platform helps to interactively design bioinformatics analysis processes. Bio-jETI is a platform for the integration, orchestration and provision of services. The agility in design and execution is demonstrated by developing seven variations on a multiple sequence alignment workflow.

Keywords: Web services, service orchestration, model-driven development, bioinformatics workflows.

1 Introduction

Bioinformatics algorithms, tools, and databases have been developed for several years now, assisting researchers in different phases of their data analyses. However, working with a large set of different tools, distributed throughout the world, is a cumbersome and error-prone undertaking when the single steps are carried out manually. Challenging bioinformatics tasks like microarray and proteomic data analyses require complex processes, and specific care in combining, monitoring, and documenting the single analysis steps. Therefore, frameworks that provide the means for automating complex bioinformatics analyses involving a number of heterogeneous services have begun to enjoy great popularity.

With *Bio-jETI* [1] we provide a comprehensive, nevertheless intuitive, graphical framework that helps biologists to integrate services, build processes (in the form of directed graphs) from the emerging components, analyze and execute them, and finally deploy and provision them as applications or services. Internally Bio-jETI uses a multi-purpose domain-independent modeling framework,

* This work has been partially supported by the Center of Applied Proteomics (ZAP) Dortmund. The EU-raised ZAP focuses on the development of new technologies in proteomics, glycoanalysis, proteinbiochips, biostatistics, and bioinformatics in terms of life science.

I. Măndoiu, R. Sunderraman, and A. Zelikovsky (Eds.): ISBRA 2008, LNBI 4983, pp. 445–456, 2008.

the *Java Application Building Center* (*jABC*) [2], for the process definition and management, and the *Java Electronic Tool Integration* framework (*jETI*) [3] for dealing with the integration and execution of remote services. Both are based on well-established software technology and have been applied successfully to different application domains.

Several workflow systems have been used in the biological domain: Taverna [4,5] and Kepler [6], but also the Bio-SPICE Dashboard [7], Triana [8,9], Pegasus [10] and VIEW [11]. Taverna and Kepler are born on top of fine-granular Grid projects: Kepler, for example, features a component for workflow definition, but at the grid-management level. As grid-based systems, Kepler and Taverna are inherently data-flow-oriented workflow systems.

Through the underlying jABC, Bio-jETI is on the contrary clearly a control-flow-oriented environment for service design and analysis. In our opinion, this is an evolution step: bioinformatics processes become increasingly networked, parallel, conditional, event-driven, recursive, and asynchronous: this is the kind of complexity sources whose control is at the core of jABCs strengths. Additionally, clear formal semantics is the precondition for a formal analysis and verification of properties of the designed workflows based on automatic mathematical proofs. The jABC has been built with this formal capability in focus, as well as with the ability to scale for large models.

The objective of this paper is to illustrate how Bio-jETI enables end users (biologists, statisticians, biochemists) that are not IT experts to define, analyze, execute, modify, and interactively develop bioinformatics analysis processes in an agile way. For convenience the examples are woven around a multiple sequence alignment, since it is commonly known and part of many analyses in genomics, proteomics, and transcriptomics. Bear in mind, however, that the presented methods are neither limited to the alignment example nor to the mentioned fields, but rather applicable whenever a service for a subtask of a complex analysis is available.

After a brief introduction of Bio-jETI's use (Sect. 2), we interactively develop a variety of processes involving a multiple sequence alignment (Sect. 3), then we elaborate on the analysis and further capabilities of Bio-jETI (Sect. 4), and finally we conclude in Sect. 5.

2 Using Bio-jETI for Service-Oriented Process Development

Bio-jETI is a sophisticated platform for integrating, orchestrating, and providing bioinformatics services [1]. Workflows in Bio-jETI are built by constructing service compositions (called Service Logic Graphs, or SLGs) that orchestrate basic services (in the form of *SIBs - Service-Independent Building Blocks*) along the flow of control.

All the user interaction happens within an intuitive graphical environment, hardly requiring any classical programming skills. Figure 1 shows the GUI: the available SIBs are listed in a browser (upper left), from where they can be

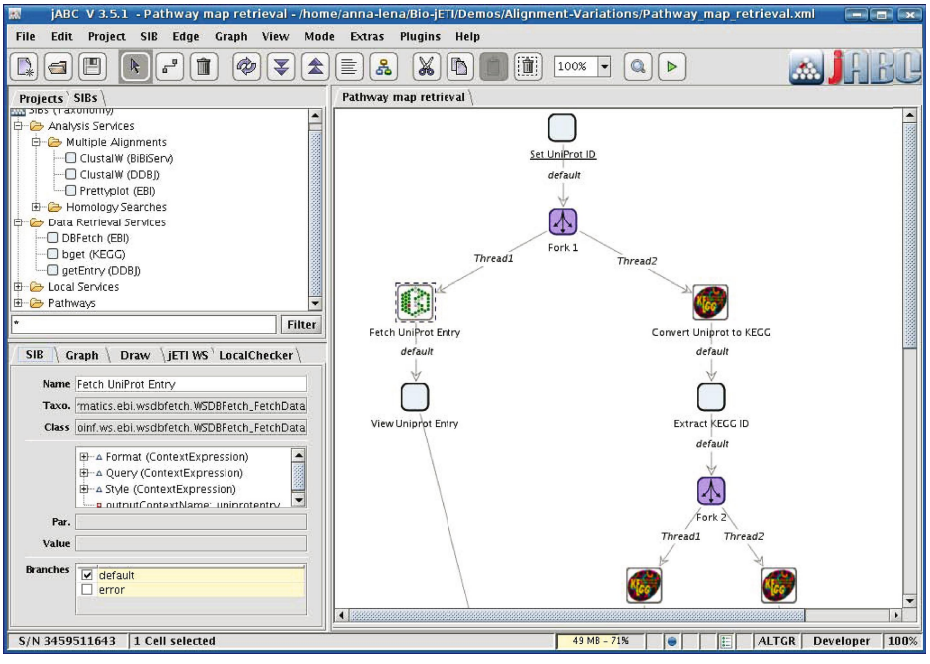


Fig. 1. The graphical user interface of Bio-jETI

dragged onto the drawing area (right), where the SLG construction takes place. Different inspectors (lower left) can be used for the detailed configuration of components and models.

The advantages of the tool are manifold: being a control-flow-oriented service definition environment, it is adequate to support complex control structures as primitives. For example, iterations over lists or matrices are provided as SIBs in the environment. At the same time the data dependencies (which are secondary to the control flow) do not clog the representation: even large processes with complex dataflows are still easily readable.

Moreover, SLGs are at the same time mathematically analyzable objects: they are directed graphs, whose nodes (the SIBs) represent basic services and whose edges (their branches) define the flow of control. They are thus amenable to the sophisticated analyses provided by modern Computer Science (see Sect. 4). We show all these features on a simple example of sequence alignment.

3 Seven Variations of an Alignment Workflow

Sequence alignments try to find correspondences between the bases or codons of DNA, RNA, or amino acid sequences. The aim is to establish similarities between sequences which result from the existence of a common ancestor [12]. One of the most popular alignment algorithms is *ClustalW* [13]. It computes

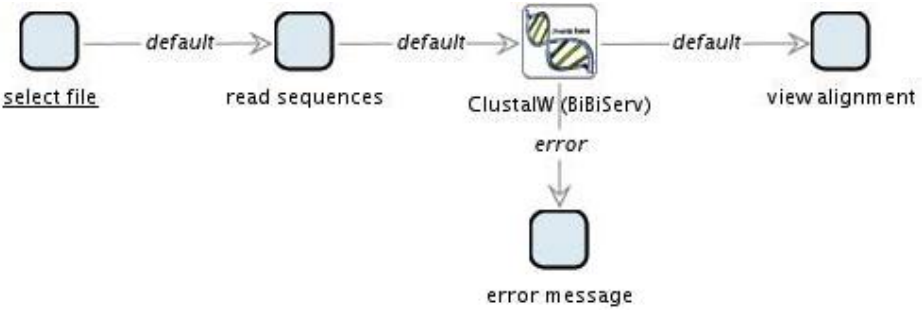


Fig. 2. Simple alignment process, using BiBiServ's ClustalW web service

fast pairwise alignments of the input sequences in order to establish a so-called guide-tree, which is then used to settle the order in which the multiple alignment is assembled from the sequences.

In the following we show along a typical workflow design session how the capabilities of Bio-jETI help discovering, testing, modifying, adapting, and customizing processes that center on two ClustalW implementations which can be accessed via web services provided by the Bielefeld University Bioinformatics Server (BiBiServ) [14] and the DNA Data Bank of Japan (DDBJ) [15], respectively. Accessing these algorithms as local Bio-jETI resources, e.g. via a local copy of a ClustalW implementation, is a simpler special case.

3.1 The Simple Alignment Process

In our first example (figure 2) the web service call to CustalW is encapsulated by the SIB with the BiBiServ icon (third from left). The surrounding process realizes some simple data management: the initial SIB (with the underlined name) selects a file from the local file system, its content is then read into the execution context, the data is sent to the web service, and finally the result is displayed to user. If an error occurs during the remote call, an appropriate error message is displayed. Due to the process surrounding the pure computation it is also possible to refine the error handling in a proactive fashion: we can, e.g., introduce fault tolerance or at least enforce a graceful termination of the computation, providing the possibility of resuming the process execution at the failed step at a later date.

3.2 Fetching the Input Sequences from a Database

The input data for the alignment is often not initially available at the local filesystem. If the desired data is, for instance, stored in the EMBL sequence database, the DBFetch web service [16,17,18] of the European Bioinformatics Institute (EBI) can be used to retrieve a set of sequences. For our example process, this means that the SIBs that read the sequences from a file must be *replaced* by SIBs that fetch the sequences via a web service and put them into

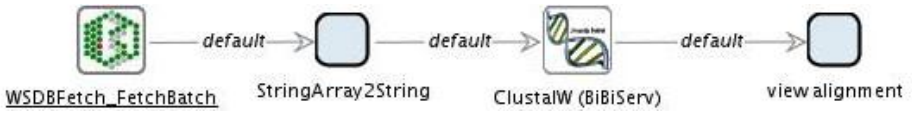


Fig. 3. Additional remote service retrieving data from a public repository

the execution context in the required format. This is done by drag and drop in the Bio-jETI GUI (figure 3). Here we need also data mediation: the SIB StringArray2String performs the necessary conversion between the two different formats. In order to facilitate reading we will omit error branches from now on.

3.3 Using a Different Alignment Service

Although the ClustalW algorithm itself is generally the same, different implementations have different characteristics, especially regarding accepted input formats, parameters, and structure of the output. While the result obtained by BiBiServ's ClustalW implementation is the pure alignment, the output of the ClustalW web service of the DDBJ is more elaborate, providing detailed statistics and a description of the implied phylogenetic tree, which may be useful for some analyses. Replacing the SIB calling BiBiServ by one that calls the DDBJ analogon yields the process displayed in figure 4. The process is immediately executable.

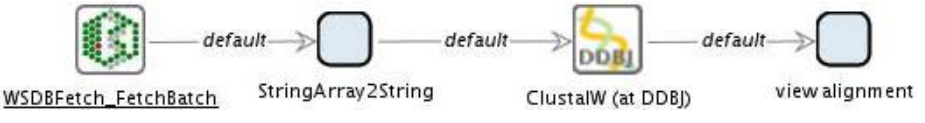


Fig. 4. Alternative alignment service: using DDBJ instead of BiBiServ

3.4 Interaction: Letting the User Chose the Service

If it is not known at process modeling time whether the pure BiBiServ alignment or the elaborate DDBJ output is wanted, it is useful to leave the choice to the user. For this purpose, an interactive SIB displays a customized message dialog at runtime, asking the user to take a decision. In this case the user can choose between DDBJ and BiBiServ (figure 5, center). The subsequent SIB checks which service has been chosen and directs the flow of control accordingly.

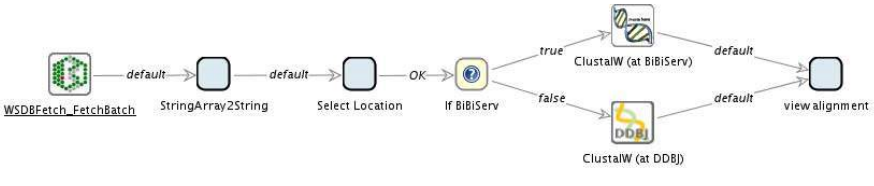


Fig. 5. User interaction and conditional branching

3.5 Visualization: The Implied Phylogenetic Tree

As mentioned before, the result of the DDBJ's alignment web service contains a description of the implied phylogenetic tree. This can be extracted via a regular expression (SIB extract tree, figure 6), written into a file and then displayed by a specific viewer, in this case ATV (A Tree Viewer [19]). Furthermore, since this visualization workflow is a quite widely reusable step, this sequence of SIBs is exported as a subprocess on its own, as shown in the bottom part of the figure.

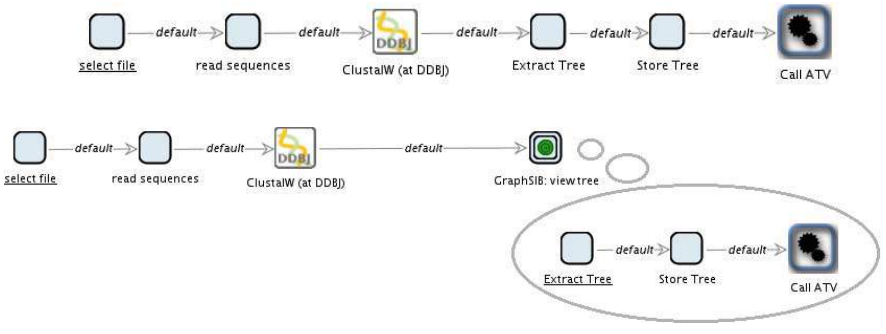


Fig. 6. Invocation of specific viewers and definition of a subprocess

3.6 Comparing Two Alignments

The biological truth is usually not known, therefore bioinformatics algorithms approximate a set of researcher-defined criteria in order to establish some hypothesis that subsequently has to be validated in further experiments. ClustalW's computations, like those of many other algorithms, can be fine-tuned by means of a number of parameters, for instance the value for the gap-open and gap-extension penalties. To find out whether our input sequences produce a stable alignment hypothesis, we can, e.g., run one instance of ClustalW with low and another with high gap penalties, and compare the results. Figure 7 shows a process that starts two parallel threads, each executing ClustalW with different parameters, and finally calls a tool which visualizes possible differences between the results, here TkDiff, a visualizer for the Unix `diff` command.

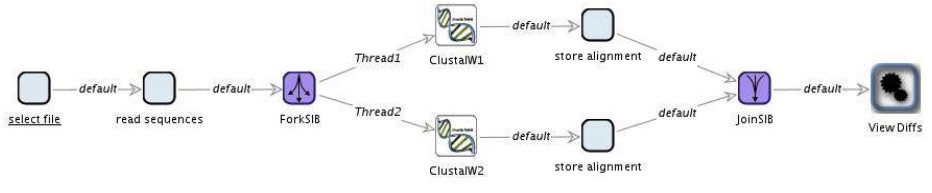


Fig. 7. Parallel threads and invocation of specific services

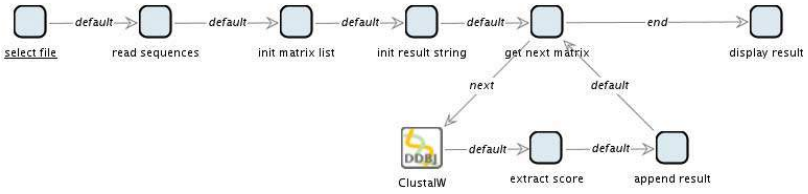


Fig. 8. Loops and result aggregation

3.7 Comparing Several Alignments

To evaluate the impact of more than two different parameters on the result a pairwise diff-viewing of all results is not a feasible method. Since alignments have a score, i.e. a number indicating how "good" the alignment is (low scores usually imply many matches, while high scores result from mismatches and gaps), this indicator can be used to obtain a survey of the results' diversity. The process pictured in figure 8 iterates over a list of substitution matrices (for instance PAM, BLOSUM, GONNET, and ID), and executes ClustalW with a different matrix parameter in each iteration. The score of each alignment is parsed out and added to a result pattern that is finally displayed.

4 Exploiting More of Bio-jETI's Potential

The processes presented in the previous sections do not, of course, cover all the modelling capabilities of Bio-jETI. The features shown in the different versions of the example can be combined, and the processes can be extended with whatever functionality is available in Bio-jETI (and encapsulated in a SIB). Our GeneFisher-P [20] gives an example of a more complex process realized with Bio-jETI: PCR primer design comprises an alignment step in case the input consists of multiple nucleic or amino acid sequences. Depending on the kind of the acutal input, services for consensus calculation, backtranslation, and finally for the primer design itself are invoked by the process.

Basic services as SIBs

The atomic actions from which Bio-jETI workflows are assembled are provided by the process building blocks, hence the potential for the processes is defined by the collection of available SIBs. The jABC provides rich SIB collections for commonly occurring tasks: SIBs that realize control-flow constructs like conditional branching, loops, and parallelism, but also libraries for working with lists and matrices and libraries for incorporating GUI elements.

What is more, any SIB available in the underlying jABC platform is reusable inside Bio-jETI. For example, messaging and telecommunication services exposed as Parlay-X web services can enhance the Bio-jETI processes: an SMS notification in case of termination or exception handling is easily added. This interdisciplinary synergy via reusal is one of the key strengths of the underlying technology.

If new building blocks are required, there are several ways to obtain new SIBs. Clients for remote services can, e.g., be generated from their WSDL descriptions by the *jETI* plugin. The *SIB-Creator* plugin generates buildings blocks that invoke methods of a specific API, like, e.g., the BioDOM [21] libraries or BioJava [22].

Orchestration without programming
Together with the sophisticated *execution context* and *hierarchy* concepts, the graphical process definition in the jABC is in fact as powerful as application development in a classical programming language. It is, however, directly accessible by non-IT experts, unlike the scripting or programming-based approaches still most common today.

Execution and compilation
Process models consisting of completely implemented SIBs are directly executable. The *Tracer* plugin allows for the overall or step-wise execution of the workflow. Figure 9 illustrates the execution of a process similar to that of figure 6. Additionally, the Tracer provides detailed information on the execution history, the objects in the context, and running threads, and is thus useful for experimenting, testing and debugging purposes.

The *GeneSys* code generator can be used to compile any of the shown executable process models into a separately deployable piece of code that can be run independently of Bio-jETI and the jABC.

4.1 Compliance to Policies

More plugins are available in order to support or enhance process development. The *LocalChecker* and the *ModelChecker* verify constraints on single SIBs and on the whole model, respectively. Users can in fact specify rules that express *policies*,

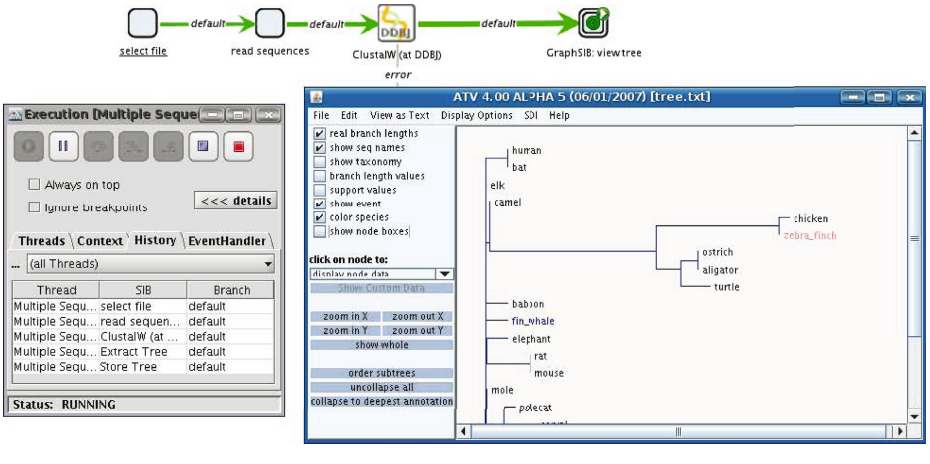


Fig. 9. Process execution with the Tracer plugin

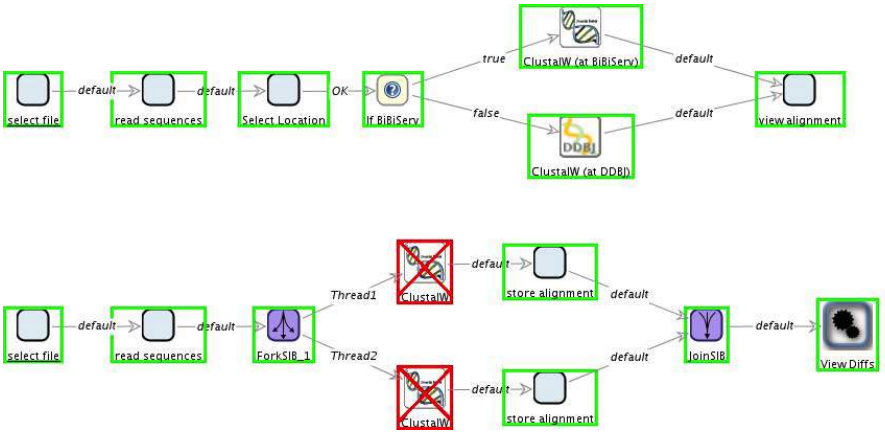


Fig. 10. Results of model checking alignment processes against policies

or *best practices*, or *constraints*. A possible policy is that the alignment location is always chosen by the user, i.e. that an alignment computation is always preceded by a location selection. In a user-friendly variant of Computation Tree Logic (CTL), a possible input language for the model checker, this is expressed as

"alignment" implies (previously "location-selection").

Figure 10 shows the result of model-checking two previous realizations against this formula. As we see, the workflow from figure 5 respects this policy (all SIBs are framed in green), while that from figure 7 does not: it does not provide any user interaction step before launching the algorithms and thus does not fulfill this requirement. The SIBs at which we detect the violation (the two algorithm executions) are marked by a red frame.

4.2 jETI

The jETI platform [3] is used within Bio-jETI to accomplish the communication with remote tools. This includes acting as a client for (SOAP and REST) web services, CORBA IDL or other RPC standards. This way the provision of appropriate SIBs happens as far as possible by generation based on a standardized service description, for instance WSDL (see figure 11). We have already successfully imported in the past entire SIB palettes for, e.g., FASTA or Muscle services. The complete deployment of applications into new web services can be realized by jETI as well.

jETI provides also a specific technology for making file-based Java or command line applications accessible via the internet. In jETI, the application provider maintains a *server* that accesses (a collection of) applications on the one side, and on the other it provides an interface to the internet. At runtime, the server receives service requests from a client (the Bio-jETI SLG) and forwards them to the actual applications, then collects the results and builds adequate

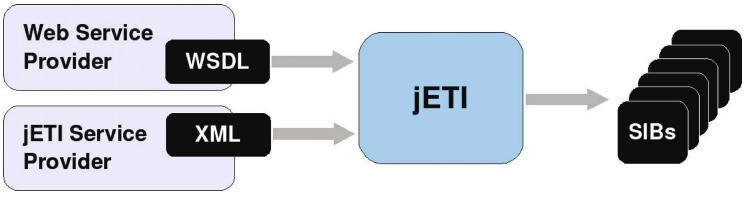


Fig. 11. Service Integration via the jETI Technology

response messages for Bio-jETI. Similar to the web services's WSDL descriptions, relevant request parameters as well as the actual calls used by the jETI server to execute the applications are defined in an XML file. This information is used by jETI to automatically generate appropriate SIBs (figure 11). Integration of services by means of jETI is convenient especially in the case of legacy application, REST services, and whenever else the setup of a classical web service is not adequate or feasible.

5 Conclusion and Perspectives

We have illustrated on a well known example, a multiple sequence alignment workflow, how Bio-jETI enables an agile way of working with analysis processes. We also showed the potential for automated analysis that the platform provides on the example of a policy rule. A flash demo, available at our web site[1], involves processes similar to those presented in this paper and illustrates more lively the interaction, the agility of the orchestration process, and the execution behaviour.

The shown alignment workflows are only one simple example. Since the underlying technologies, the jABC and jETI, have been designed in a domain-independent fashion, any algorithm, tool, or database can be integrated, provided that there is a way of accessing it programmatically. In fact, previous and current Bio-jETI projects address diverse life science disciplines, including complex database searches, sequence-based processes like PCR primer design [23] or retrieval of orthologous IDs [24] as well as statistical analysis of LC/MS experimental data using GNU R statistics packages [25] and network analysis and visualization with Cytoscape [26].

Current work on Bio-jETI comprises, among others, a comprehensive integration mechanism for GNU R statistics packages and a plugin for automated service discovery based on the Bio-MOBY ontologies in order to increase the range of services that can be integrated into the platform in a fully automated fashion. Furthermore, we plan to enhance Bio-jETI with workflow synthesis techniques [27] that we have already successfully applied within other projects, such as the Semantic Web Service Challenge [28].

[1] `http://jeti.cs.uni-dortmund.de/biojeti/downloads/biojeti_wink.html`

References

1. Margaria, T., Kubczak, C., Steffen, B.: Bio-jETI: A Service Integration, Design, and Provisioning Platform for Orchestrated Bioinformatics Processes. BMC Bioinformatics (to appear)
2. Jörges, S., Kubczak, C., Nagel, R., Margaria, T., Steffen, B.: Model-Driven Development with the jABC. In: Bin, E., Ziv, A., Ur, S. (eds.) HVC 2006. LNCS, vol. 4383, Springer, Heidelberg (2007)
3. Margaria, T., Nagel, R., Steffen, B.: jETI: A Tool for Remote Tool Integration. In: Halbwachs, N., Zuck, L.D. (eds.) TACAS 2005. LNCS, vol. 3440, pp. 557–562. Springer, Heidelberg (2005)
4. Oinn, T., Addis, M., Ferris, J., Marvin, D., Senger, M., Greenwood, M., Carver, T., Glover, K., Pocock, M.R., Wipat, A., Li, P.: Taverna: A tool for the composition and enactment of bioinformatics workflows. bioinformatics 20(17), 3045–3054 (2004)
5. Hull, D., Wolstencroft, K., Stevens, R., Goble, C.A., Pocock, M.R., Li, P., Oinn, T.: Taverna: A tool for building and running workflows of services. Nucleic Acids Research 34(Web-Server-Issue), 729–732 (2006)
6. Altintas, I., Berkley, C., Jaeger, E., Jones, M., Ludäscher, B., Mock, S.: Kepler: An Extensible System for Design and Execution of Scientific Workflows. In: SSDBM, pp. 423–424 (2004)
7. Garvey, T.D., Lincoln, P., Pedersen, C.J., Martin, D., Johnson, M.: BioSPICE: Access to the Most Current Computational Tools for Biologists. OMICS - A Journal of Integrative Biology 7(4), 411–420 (2003)
8. Majithia, S., Shields, M.S., Taylor, I.J., Wang, I.: Triana: A Graphical Web Service Composition and Execution Toolkit. In: Proceedings of the IEEE International Conference on Web Services (ICWS 2004), pp. 514–524. IEEE Computer Society, Los Alamitos (2004)
9. Taylor, I., Shields, M., Wang, I., Harrison, A.: The Triana Workflow Environment: Architecture and Applications. In: Taylor, I., Deelman, E., Gannon, D., Shields, M. (eds.) Workflows for e-Science, Secaucus, NJ, USA, pp. 320–339. Springer, New York (2007)
10. Deelman, E., Singh, G., Su, M.-H., Blythe, J., Gil, Y., Kesselman, C., Mehta, G., Vahi, K., Berriman, G.B., Good, J., Laity, A.C., Jacob, J.C., Katz, D.S.: Pgasus: A framework for mapping complex scientific workflows onto distributed systems. Scientific Programming 13(3), 219–237 (2005)
11. Chebotko, A., Lin, C., Fei, X., Lai, Z., Lu, S., Hua, J., Fotouhi, F.: View: A visual scientific workflow management system. In: IEEE SCW, pp. 207–208. IEEE Computer Society, Los Alamitos (2007)
12. Polanski, A.: Sequence Alignment. In: Bioinformatics, pp. 155–156. Springer, Heidelberg (2007)
13. Thompson, J.D., Higgins, D.G., Gibson, T.J.: CLUSTAL W: Improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. Nucleic Acids Res. 22(22), 4673–4680 (1994)
14. Hartmeier, S., Krüger, J., Giegerich, R.: Webservices and Workflows on the Bielefeld Bioinformatics Server: Practices and Problems. In: Proceedings of NETTAB 2007 Workshop: A Semantic Web for Bioinformatics (2007)
15. DDBJ: Web API for Bioinformatics (2007), `http://xml.nig.ac.jp/wsdl/index.jsp`

16. Pillai, S., Silventoinen, V., Kallio, K., Senger, M., Sobhany, S., Tate, J., Velankar, S., Golovin, A., Henrick, K., Rice, P., Stoehr, P., Lopez, R.: SOAP-based services provided by the European Bioinformatics Institute. Nucleic Acids Research 33(1), 25 (2005)
17. Labarga, A., Pilai, S., Valentin, F., Anderson, M., Lopez, R.: Web services at the European Bioinformatics Institute. EMBnet.news 11(4), 18–23 (2005)
18. Labarga, A., Valentin, F., Anderson, M., Lopez, R.: Web Services at the European Bioinformatics Institute. Nucleic Acids Research Web Server Issue (2007)
19. Zmasek, C.M., Eddy, S.R.: ATV: Display and manipulation of annotated phylogenetic trees. Bioinformatics 17(4), 383–384 (2001)
20. Lamprecht, A.-L., Margaria, T., Steffen, B.: GeneFisher-P: Variations of GeneFisher as Processes in Bio-jETI. In: Proceedings of NETTAB 2007 Workshop: A Semantic Web for Bioinformatics (2007)
21. Seibel, P.N., Krüger, J., Hartmeier, S., Schwarzer, K., Löwenthal, K., Mersch, H., Dandekar, T., Giegerich, R.: XML schemas for common bioinformatic data types and their application in workflow systems. BMC Bioinformatics 7, 490 (2006)
22. BioJava Project: Main Page - BioJava, `http://biojava.org`
23. Lamprecht, A.-L., Margaria, T., Steffen, B., Sczyrba, A., Hartmeier, S., Giegerich, R.: GeneFisher-P: Variations of GeneFisher as Processes in BiojETI. BMC Bioinformatics (to appear)
24. Margaria, T., Kubczak, C., Njoku, M., Steffen, B.: Model-Based Design of Distributed Collaborative Bioinformatics Processes in the jABC. In: Proc. ICECCS 2006, IEEE International Conference on Engineering of Complex Computer Systems (August 2006)
25. Kubczak, C., Margaria, T., Fritsch, A., Steffen, B.: Biological LC/MS Preprocessing and Analysis with jABC, jETI and xcms. In: Proceedings of the 2nd International Symposium on Leveraging Applications of Formal Methods, Verification and Validation (ISoLA 2006), Paphos, Cyprus, pp. 308–313. IEEE Computer Society Press, Los Alamitos (2006)
26. Shannon, P., Markiel, A., Ozier, O., Baliga, N.S., Wang, J.T., Ramage, D., Amin, N., Schwikowski, B., Ideker, T.: Cytoscape: A Software Environment for Integrated Models of Biomolecular Interaction Networks. Genome Research 13, 2498–2504 (2003)
27. Margaria, T., Steffen, B.: Ltl guided planning: Revisiting automatic tool composition in eti. In: SEW, pp. 214–226. IEEE Computer Society, Los Alamitos (2007)
28. SWS Challenge: Challenge on Automating Web Services Mediation, Choreography and Discovery, `http://sws-challenge.org/`

Supporting Computational Systems Science: Genomic Analysis Tool Federations Using Aspects and AOP

David Stotts, Keith Lee, and Ivan Rusyn

Department of Computer Science, and School of Public Health
University of North Carolina at Chapel Hill
{stotts,lee}@cs.unc.edu, iir@unc.edu

Abstract. We show how Aspect-Oriented Programming (AOP) and its main concept – the aspect – can be used to effectively construct interoperating collections of scientific tools and models. Such collections, termed "federations", naturally arise in computational frameworks for bioinformatics problems. Programming modern scientific simulations and models require more domain expertise than can be found in one researcher; often the many researchers needed to create the various computational components of a full solution cannot be gathered to work as a single controlled software development team. Our approach allows individuals to construct their own components and tools, and then have them assembled without alteration (and without coordination of the original programmers) into a federation for the larger final computational solutions. We illustrate the methods with two SNP and haplotype analysis tools written in Python.

1 Systems Science and Model Federations

This paper demonstrates how a new programming technology -- aspect-oriented programming -- can assist in the creation of modern scientific software, the kind needed for systems science. In this first section we discuss the nature of modern scientific software. In the next section, we give a brief overview of aspect-oriented programming. Following that, we introduce our driving problem in systems toxicology, and show how AOP allows a good solution to our needs. We conclude with a comparison of our work with prior related research projects.

Software Construction by Multiple Independent Researchers

Unlike the past, where a single science researcher could construct a model representing the physical aspects of some phenomenon under study, modern scientific software is becoming a collective effort of many different scientists working in different knowledge domains with different bodies of expertise. Though the goal is to produce a single system that simulates, explains, or characterizes some physical phenomenon, the knowledge needed to complete the model is more than can be obtained from single researchers.

Moreover, from a purely software point of view, different researchers will work individually in different programming languages, using different domain or community

I. Măndoiu, R. Sunderraman, and A. Zelikovsky (Eds.): ISBRA 2008, LNBI 4983, pp. 457–468, 2008.

libraries, on different computing platforms. All these technical factors confound efforts to assemble the individual efforts of several scientists into a cohesive explanatory and predictive model of some large complex system.

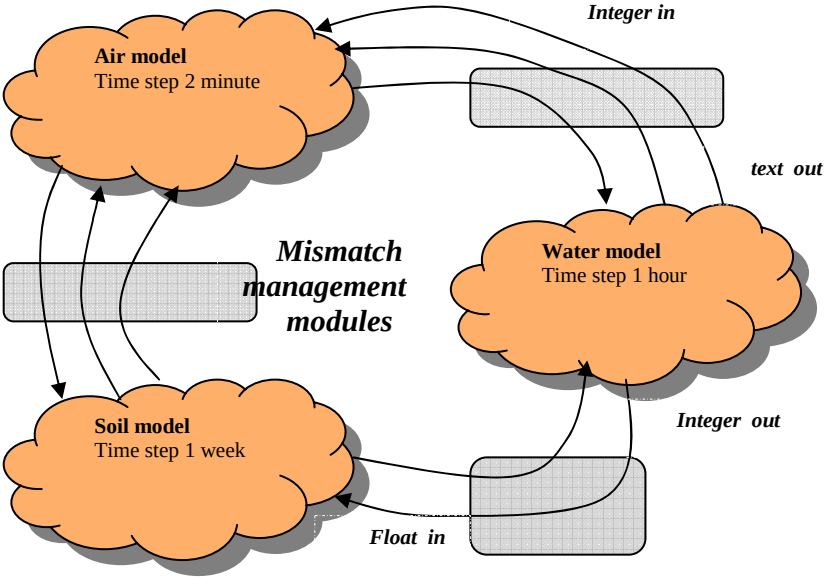


Fig. 1. Model federation, with mismatched component models

Module Mismatch in Federations

We use the term *federation* for a collection of programs developed by individual scientists, and assembled to function interoperably as a cohesive scientific model. Model federations are how future scientific software systems will function, to overcome the inherent complexity of the encoded knowledge needed for large-scale modeling, as well as to allow individuals to work semi-independently in their own domains while contributing to a purposive whole (the federation). Scientists must collaborate to get their individual contributions assembled correctly; this collaboration is often difficult, over distance. Therefore a solution to assembling federations that minimizes this coordination is desirable.

Individual models that are to execute together will mismatch in several ways, inhibiting their assembly for interoperation. First, they may be developed over different spatial frameworks and the divisions or units of space may mismatch in size, granularity, locations, etc. Next, the model simulations may operate on different time scales, giving temporal mismatch. One simulation may proceed in iteration steps that represent hours, and the output of this simulation may be needed as input to one that has an iterative time step representing seconds. Thirdly, models may exhibit mathematical mismatch in that each may employ different solution techniques… regular grids, finite element meshes, Markov chains… it may not be clear how the results of one technique can be applied to another. Moreover, error will accumulate in any

model and interconnecting models presents the problem of tracking errors accumulations across these mismatch boundaries. Figure 1 illustrates this idea of module mismatch in a federated model.

Models may also mismatch in technical ways related only to the computational platform they are constructed for. One may read and write data files and may save results internally to be emitted as a lump when the entire simulation is complete. Another may be using operating system streams and emitting data on each iteration of a simulation loop. Yet another may be using message passing in an object-oriented architecture with persistence, or saving data in a database. Interconnection of models with technical mismatches requires the intervention of a programmer for each connection, or an interconnection system which has been developed to automatically handle the common types of mismatches that occur by setting up pre-written filters. Our approach leans to the former (programmer intervention) but use of aspects means the connection code can be developed externally without altering the source code of the individual tools and models.

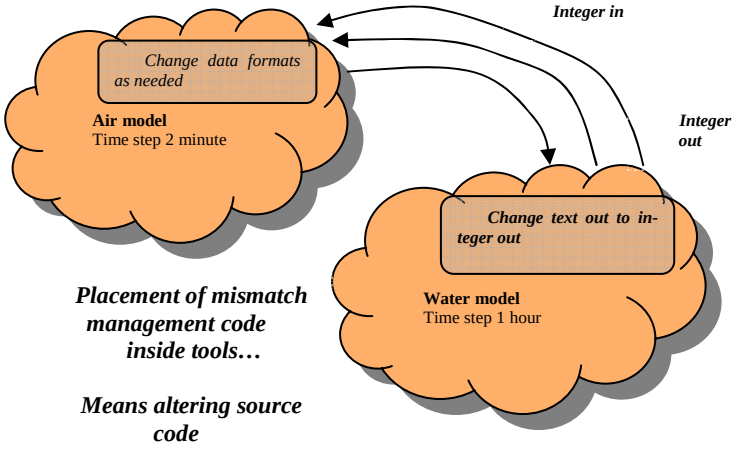


Fig. 2. Altering tool source code is a less attractive approach

Figure 2 shows one part of the model federation from Figure 1. Here, however, module mismanagement is handled with custom code that has been added to the source code of the component tools. Under this approach, federations are constructed by rewriting sections of each tool that wishes to import or export data so that format transformations are properly done. We consider this approach inferior to the aspect-based approach show in Figure 1 for several reasons. First, tool source code has to be available for alteration (the aspect-based approach requires only knowledge of method names -- information that is often available in API documentation). Secondly, this approach may require rewriting a tool differently for each different combination in which the tool is needed. Using aspects means writing different aspects for different federations, but these aspects are separate code modules and no change is needed to the tool source code itself.

Systems Science: Computational Infrastructure for Large Problems

Model federations are one example of what is becoming known as *systems science*. A computational systems approach to large-scale scientific problems is currently viewed as necessary for success in domains where many different experts are needed to assemble the sum total of the information that problems require; for example, computational systems biology seeks to model biological entities from molecular interactions, up through cellular processes, up to macro-scale structure like muscles, organs, systems, and entire organisms [11]. Inter-connecting the many component models and simulations is a very challenging computing research problem. Waters and Fostel have described similar needs and similar potential solutions for the field of systems toxicology [12].

We have studied this problem in the past in the context of environmental models for the US EPA. Our solution was based on a custom-designed framework called Deco [3], and required programming model components into the base functional programming language of Deco (Haskell). While we were able to successfully combine several models into a federation modeling the hydrology of the Neuse River [3], we found the need to change the source code of each component model a serious problem for constructing federations from the work of independent scientists working in different programming languages. We now find Aspect oriented programming to be more promising as a means of generating federated systems science solutions.

Driving Problem: Mouse Liver Toxicology Studies

Our driving problem aims at integration of genetics, toxicogenomics and conventional toxicity endpoints into a systems biology approach using mouse models for improved characterization of toxicity pathways, discovery of new molecular and cellular indicators of exposure and outcome, better dose-response assessment and more accurate inter-individual/cross-species extrapolations. Acetaminophen (APAP) was selected as a model toxicant because, despite the existence of a large amount of information regarding the mechanisms of action, conventional clinical biomarkers largely fail to connect toxicity with clinical outcomes. Furthermore, no biomarkers exist for predicting toxicity before standard signs of toxicity are observed, or for determining individual susceptibility to APAP overdose. The conventional toxicology and the new -omics data from this study are being correlated with knowledge of genetic differences between mouse inbred strains with an underlying goal to identify predictors of genetic predisposition to toxicant-induced organ damage.

We have designed and built computational approaches that improve the linkage in the source-to-outcome paradigm by automating and streamlining the data flow from each of these methods that contribute to the definition of parts of the metabolic and gene expression control networks that predict the toxicity outcomes and define the susceptible genotypes. Figure 3 shows how these analyses require multiple data streams and sources to be fed as input to numerous tools, and the output of tools used respectively as input to tools further down the line. In many cases, the data must be transformed, edited, altered, or enhanced for it to match the input requirements for the tools being used. This process has been done manually in the past, and is both time consuming and error prone.

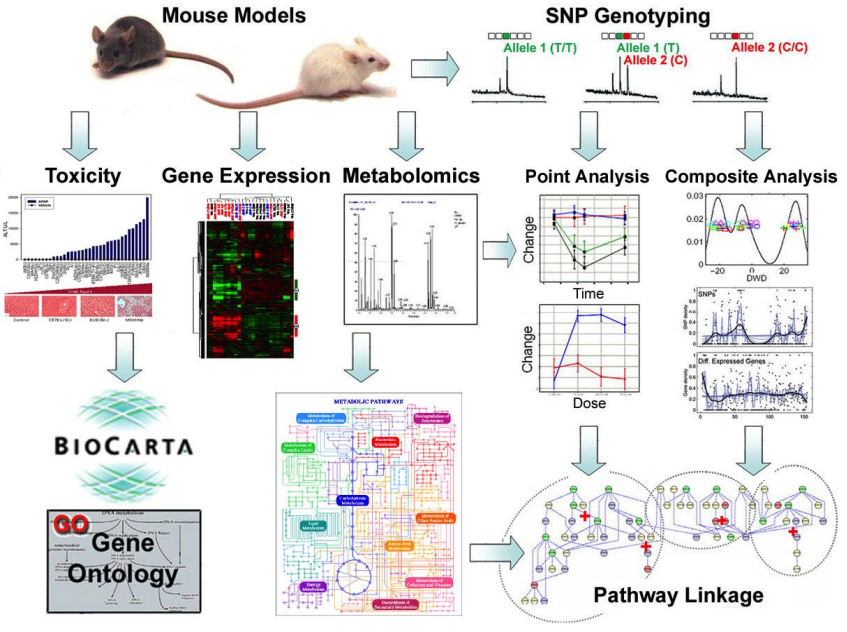


Fig. 3. Workflow among tools required for mouse tox studies

Our AOP-based solution is a framework that creates filters to arbitrate data flow among the various tools. Automating this currently-manual construction makes it far less error prone and more easily repeatable. The framework effectively allows the tools to interoperate, producing and consuming data that is massaged correctly by the filters into the required formats and structure. These "filters" are in reality aspects written in AOP-enhanced languages such as AspectJ [7] for Java, and the SpringPython system [3] for Python. We have successfully integrated several of the tools in the workflow, and are currently working to complete the suite. Our work so far has been to prove the principle of using aspects for weaving together federations, and we the results are encouraging.

Several of the tools in our workflow are not written in Java or Python. Or next step in integration will be to move to an AOP that is meant to handle multiple different source languages. There are several research efforts underway on AOP notations for weaving together multiple source languages, but none is yet as production ready as AspectJ or SpringPython.

In the following sections we first give a more through explanation of aspect oriented programming, and then we illustrate the aspect-based method for creating tool federations with a specific example using two SNP analysis tools written in Python.

2 Aspect Oriented Programming

Aspect oriented programming (AOP) [4,5] is a fairly recent development in the field of software engineering, and it offers a solution to the problem just described. The

concepts are not specific to any one language; several aspect languages are in wide use, including the AspectJ and Spring Python implementations that we use.

AOP mainly deals with cross-cutting concerns that don't fit well with the traditional programming paradigm. Typically there are levels of separation and encapsulation within a program hierarchy. Cross-cutting concerns touch multiple and varying modules within an application, often dealing with an aspect of the program that cannot be concisely captured through traditional programming means. A classic example is logging; this is a concept used by all modules and objects yet owned by none. Furthermore, making a change in logging code would mean touching many pieces of software which are otherwise unrelated.

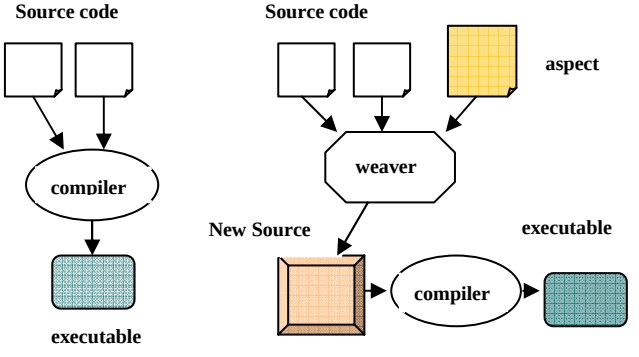


Fig. 4. Traditional programming (L) vs. Aspect oriented programming (R)

Figure 4 shows how aspects relate to traditional programming. An aspect is a separately written code module that describes several things: (1) what points in execution of the host program are the "breakpoints" at which the aspect needs to awaken and take action (2) what are the computational behaviors to be executed when the host program reaches one of the breakpoints. When the needed aspects have been written, a compiler "weaves" the aspect source code into the host source code, producing a new system for compiling and execution. Thus the host source code is not re-written by the software developer… instead, the aspect code is inserted into it automatically.

Aspects can execute before or after method calls. They can intercept the parameters that are being sent to a method, and can keep copies for other use, or pass them on to the method unchanged, or change the parameters before passing them.

Aspects as mismatch management modules in federations

Our use of aspects, however, is not for crosscutting concerns that the concept was originally developed for. Rather, we see aspects as a natural "glue" that allows data being manipulated in one program to be intercepted, drawn out, and sent (perhaps filtered and transformed) to another program, and *vice versa*. In some sense, our use of aspects is a small subset of the potential uses they have in overall programming.

We use aspects to look for input/output statements in the tools and models of a federation. Most scientific models operate by reading data files, and writing data files

of results. We key on these I/O points in order to capture the data being generated by one component model, and send that data to another model that needs it as input. The aspect, being a small program filter, can also massage the data it has captured. An aspect can transform data formats before sending it on; it can buffer data for simulations that are running at different cycle times; it can interpolate data values in order to feed an output model that needs more data than its input model is generating; an aspect can act as a data sink for an output model that is producing more data than needs to be passed on. In this sense, aspects are the mismatch management modules shown in Figure 1.

Aspects essentially become custom filters, able to be generated for the specifics of a particular federation without alteration of the base tools. Aspects can work on an entire data set as a batch, thus matching the capabilities of sequential techniques. Alternately, aspects can work in real-time as data is generated, filtering data generated by source tools before it is processed by sink tools. Aspects used in this fashion are a glue binding independent tools together without their knowledge. It allows the tools to remain independent. As the bridge technology the aspect contains and combines the information to handle data communication between data sources and data sinks. Aspects can be passive, active and reactive. Aspects work with multiple dependency path tools chains, and thus are not limited by a restrictive linear dependency chain.

3 Example: Aspects for Python SNP Tools

We can now illustrate these ideas with a combination of two existing bioinformatics applications analyzing single nucleotide polymorphisms (SNPs). The first application is called *Genome SNP Interval Analyzer* (GSIA); the second is *SNP Evolutionary Tree Analyzer* (SETA). Both were written in Python by researchers at UNC Chapel Hill, but at different times. SETA was designed to analyze the kind of data that GSIA produces, but users have been manually editing GSIA output files to make then compatible with SETA input expectations. This repetitive and error-prone practice makes these two tools ideal for AOP-based interconnection.

Between any two individual DNA sequences within a species, approximately 99.9% of the sequence is identical. The remaining 0.1% of variations accounts for the differences between all individuals. Evolutionary forces such as mutation and recombination work to create these differences. Scientists analyze the haplotype (sequence of SNPs from a single specimen) to determine relatedness between the sequences and thus which variations are due to random changes (mutation) and which are due to reproduction (recombination). Typically, an m^2 pair-wise *compatibility matrix* (m = number of SNPs) is created to do this analysis. With haplotypes stacked on each other as rows, the columns become SNPs (see Figure 5). The compatibility matrix is the pairing of these SNP columns, such that pairs that differ through recombination are marked accordingly. With large genome datasets in the gigabyte range due to dense sampling of the genome, fast algorithms are needed to produce these analyses.

(1) GSIA. The compatibility matrix is used to find regions within the genome where there is no evidence of recombination; there are many such regions within a matrix. A non-optimal linear algorithm for finding a region is to scan left-to-right. Starting with the first SNP, maintain a queue of SNPs. Moving left-to-right, compare

the next SNP with all SNPs in the queue. If there is an incompatibility due to recombination, close the interval, clear the queue, and begin the queue with the SNP that just created the incompatibility. Visually, this can be seen as a triangle starting the far left-hand corner. The triangle grows until it touches a marked block. The SNPs under this triangle are an interval. The left vertex of the next triangle begins at the column next to the right vertex of the previous triangle. This solution is not unique, since the same process can also be done right-to-left. The solution is not optimal either, as it merely finds a minimal set of non-overlapping intervals. Even though left-right and right-left are not maximal, they both have *k* intervals. The main goal is to find the minimal set of *k* maximally sized intervals.

An $O(N^2)$ algorithm would find and combine the maximum left-right and right-left intervals for each SNP. GSIA finds this set of all intervals of maximum size with linear runtime. It then uses this to find the minimal sized set of maximal intervals. Since searching through all such sets is exponential, GSIA uses an algorithm to find a set (not necessarily unique) in expected linear time, *O(k)*.

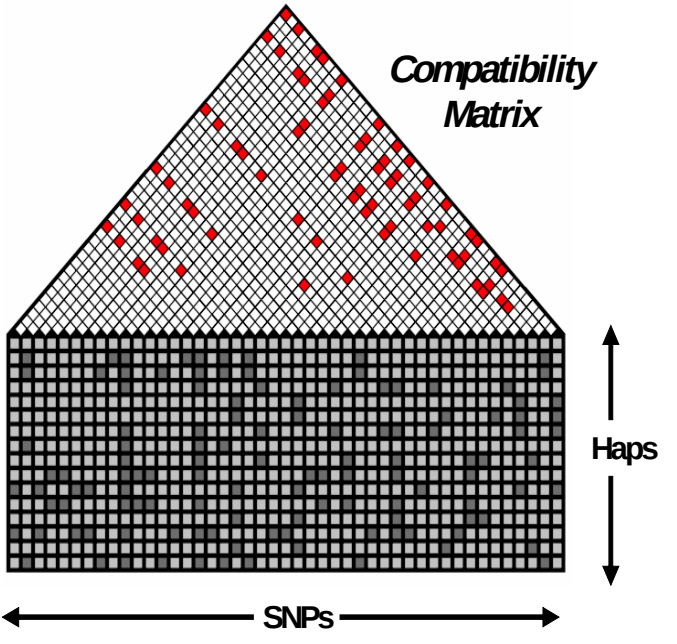


Fig. 5. Data structure analyzed by combining GSIA and SETA

(2) SETA. SETA uses SNP data to generate a phylogenetic tree of the data. A phylogenetic tree is a way to represent relatedness between sequences. Vertices within the tree are haplotypes, and the edges connecting trees represent mutations. A phylogenetic tree represents an evolutionary history of the haplotypes. It is a representation of the common ancestry betweens individuals within a species. By using the output of GSIA as the input for SETA, the data and ancestry can be represented using the fewest possible number of trees.

(3) GSIA → SETA federation. There is a simple and natural combination of these two applications. GSIA writes its results of a minimal set of *k* maximally sized intervals to a data file, which is consumed as input by SETA. However, as the tools were independently written, the data formats mismatch. GSIA writes to comma separated value files (csv); SETA expects input as text. The conversion is fairly simple (as noted above, the researchers were doing it manually with an editor): for each line, copy the first two comma-separated columns then append seven comma-separated zeros, omitting the column titles/labels. GSIA produces many csv output files, but SETA needs only a single specific one of these.

We federate these two tools by writing an aspect in SpringPython [3]. The aspect is a separate code module which SpringPython will execute concurrently with the tools; this approach does not require altering any source code in either tool. There are several strategies one can take in writing these aspects. The approach we prefer is to intercept the output actions in the data producing tool (GSIA here). Once one triggers, the aspect intercepts the data and copy/converts it to pass on to SETA (via invocation of corresponding SETA input methods). Here is the Python code for the interceptor and the action taken on interception:

```
#     ASPECT TRIGGER SPECIFICATION
#
from springpython.aop import *

# intercept the function Chromosome.printInterval()
# that writes data to csv files
pointcutAdvisor = RegexpMethodPointcutAdvisor (
   advice = [AspectOutputStreamIntercept()],
   patterns = [".printInterval.*"]
   )
service = ProxyFactoryComponent (
   target = Chromosome(), interceptors = pointcutAdvisor
   )

#     INTERCEPTOR METHOD DEFINITION
#
from springpython.aop import *

class AspectOutputStreamIntercept(MethodInterceptor):
  def invoke(self, invocation):
    #let function proceed as usual
    results = invocation.proceed()

    #determine if this is the file we are interested in
    #the following assumes that argument order is fixed
    if(invocation.args[-1].name.endswith("min_intervals.csv")):
      textFilename = invocation.args[-1].replace(".csv", ".txt")
      #filter some of the data intended for csv file into text file
      textFile = file(textFilename, "w")
      #use the data values a & b (args a & b are the second and
      #third argument respectively), followed by seven 0s
      textFile.write(args[1] + "," + args[2] + ",0,0,0,0,0,0,0")
      textFile.close()

    #return the original function
    return results
```

This aspect targets the function *printInterval()* called in GSIA that actually does the csv file writing. *printInterval* is passed values and a file, and writes those values to the file. We designed the class *AspectOutputStreamIntercept* to run on the trigger. This interceptor determines if this is the file we are interested in, and if so copies the target data to the text file in the format that SETA uses.

This strategy for writing the aspect makes data available to SETA line by line, which can be useful if the target tool can work effectively with input data coming in piecemeal. Given the code structure of GSIA, we could also have written an aspect to key on a different method call – specifically, the enclosing one that generates the entire output file. Under this strategy, no data would be made available to SETA until all data was produced in GSIA. The net computational results would be the same, but the runtimes would vary due to forced sequential tool executions. The piecemeal approach will be most useful when tool execution times are very lengthy. Creating federations via aspects does require design thinking like producing any software.

Another strategy entirely for writing federating aspects is to key on method executions in the *receiving* tool (SETA here). We have produced examples where execution of the data input functions in SETA cause an aspect to trigger; the aspect then seeks the output data produced by GSIA and converts it from csv to text. Which strategy to use (aspect pushes data vs. aspect pulls) is a design issue that has to be determined by tool synchronization needs, access to source code (we don't alter source code, but we do have to know the method names to define triggers), etc.

Another interesting and useful feature with aspects is the ability to decide in the aspect code whether to allow the triggering method call (such as the file write in GSIA) to complete normally or not. In our example, this means we could allow the producing tool (GSIA) to write its output data as normal (which we did) or we could shunt execution back to the tool without completing the file write (after the data was copy/converted out to the receiving tool SETA). All these options are achieved in the aspect code without any alteration of the tool source code.

4 Related Prior Research

Software researchers have been developing methods for "programming in the large" for years, allowing different programs written in different languages to be combined into cohesive computational engines. The area is collectively known as *middleware*. The literature is too large to survey here, but it is distinct from the AOP approach we have described in that middleware is usually dependent on a custom-built framework or software platform that requires custom programming of each component module. The AOP approach allows independent model development and assembly without coordination of the various researchers contributing to a federation.

By thinking of model observation as a cross-cutting concern, aspects can be used to create observers and allow the models to be created independent [1]. They note that the computer simulation language MAML, designed to help scientists in a domain outside of computer science develop computer simulations, has some AOP functionality already built-in. Model observers created as aspects are simpler to code and more

intuitive to understand. A modeling language supporting AOP further simplifies understanding and development of both model and observer. Adding more AOP features to MAML can improve and enhance construction of computer simulations.

AOP has been proposed as a way to separate the two core concerns of high performance scientific computing; the mathematical model and its parallel execution [2]. There are various obstacles of varying difficulty to achieving this goal. Current techniques for parallelization such as for-loop iteration and compiler directives are not aspect friendly, plus code may not follow good software design technique. Re-factoring models to improve the base code design and support use of aspects is possible, but a more solid underlying base code design introduced early in the life cycle will best exploit the benefits of AOP. More study is also needed on higher dimension tasks.

Adaptation of middleware technology has increased as such systems can be used for wide general use, or for optimized specific use. As their functionality and complexity increase, middleware following traditional software methodologies has become harder to use and configure. Aspects can be factored out of legacy middleware systems to reduce complexity and improve both performance and configurability [8]. This allows them to be general use without a large footprint or specialized without multiple code versions.

In earlier work addressing how to handle the complexity and footprint issues with feature rich middleware, aspects are used to incrementally add features [9]. To get flexible and extensible middleware, features are conceptualized as combinations of aspects. The code base contains the common core features. With the user selecting the feature they need, there is a level of separation between the user and the interdependency of its aspects. Aspect consistency is important; feature selection must account for things like missing dependencies and conflicting features. With this large number of possible aspect configurations, automated testing becomes even more critical. Aspects can also be used to create simple yet thorough unit testing.

Increased flexibility and simplified programming are not the only benefits of using aspects in middleware; consistency across product-lines and improved scalability are also sighted as benefits [10]. One group of concerns, homogenous concerns across product-lines, include tracing and logging, first failure data capture for error analysis and reporting, plus capturing monitoring and statistics data. With those concerns implemented fairly consistently in all applications, large teams using them through aspects can reduce policy and implementation costs. Heterogeneous concerns, with behavior changing depending on its location in the code, need aspect-oriented refactoring to provide the best benefit. The aspects can be easily integrated into the build of large scale applications with minimal overhead on the compilation time.

Acknowledgements

This research has been supported by the United States Environmental Protection Agency under a grant # STAR-RD832720 to the University of North Carolina at Chapel Hill, creating the Carolina Bioinformatics Research Center.

References

[1] Gulyas, L., Kozsik, T.: The Use of Aspect-Oriented Programming in Scientific Simulations. In: Proceedings of Sixth Fenno-Ugric Symposium on Software Technology, Estonia (1999)
[2] Harbulot, B., Gurd, J.: Using AspectJ to Spearate Concerns in Parallel Scientific Java Code. In: Proceedings of the Third International Conference on Aspect-Oriented Software Development (AOSD), Lancaster, UK (March 2004)
[3] SpringPython, a cross-platform programming framework with aspects, http://springpython.python-hosting.com/wiki/AspectOrientedProgramming
[4] Aspect Oriented Software Association, Aspect-oriented software development (October 2006), http://aosd.net/
[5] Laddad, R.: I want my Aspect Oriented Programming!, JavaWorld.com (January 8, 2002), http://www.javaworld.com/javaworld/jw-01-2002/jw-0118-aspect.html
[6] Herington, D., Stotts, D.: DeCo: A Declarative Coordination Framework for Scientific Model Federations. In: Proc. of the IEEE Conf. on Automated Software Engineering 2003, Montreal, October 6–10, 2003, pp. 60–69 (2003)
[7] Eclipse Project, AspectJ: Crosscutting Objects for Better Modularity (December 2006), http://www.eclipse.org/aspectj/
[8] Zhang, C., Jacobsen, H.: Quantifying Aspects in Middleware Platforms. In: 2nd International Conference on Aspect Oriented Systems and Design, Boston, MA, pp. 130–139 (March 2003)
[9] Hunleth, F., Cytron, R., Gill, C.: Building Customizable Middleware using Aspect Oriented Programming. In: Workshop at OOPSLA (2001)
[10] Colyer, A., Clement, A.: Large-scale AOSD for Middleware. In: Proceedings of the Third International Conference on Aspect-Oriented Software Development (AOSD), Lancaster, UK (March 2004)
[11] Kitano, H.: Computational Systems Biology. Nature 420, 206–210 (2002)
[12] Waters, M.D., Fostel, J.M.: Toxicogenomics and Systems Toxicology: Aims and Prospects. Nature Reviews Genet. 5, 936–948 (2004)

BioDQ: Data Quality Estimation and Management for Genomics Databases

Alexandra Martinez*, Joachim Hammer, and Sanjay Ranka

Dept of Computer & Information Science & Engineering, University of Florida, Gainesville, FL 32611, USA
{amartine, jhammer, ranka}@cise.ufl.edu

Abstract. We present BIODQ, a model for estimating and managing the quality of biological data in genomics repositories. BIODQ uses our Quality Estimation Model (QEM) which has been implemented as part of the Quality Management Architecture (QMA). The QEM consists of a set of quality dimensions and their quantitative measures. The QMA combines a series of software components that enable the integration of QEM with existing genomics repositories. The basis of our experimental evaluation is a research study conducted among biologists. Evaluation results show that the QEM dimensions and estimations are biologically-relevant and useful for discriminating high quality from low quality data. The most relevant capabilities of the QMA are also presented.

Keywords: Data Quality, Genomics Databases, GenBank, RefSeq, quality dimension, measure, estimation, management, classification, architecture.

1 Introduction

The rapid accumulation of biological information as well as their widespread usage by scientists to carry out research is posing new challenges to determine and help manage the quality of data in public genomics repositories. Genbank [1], RefSeq [2], and Swissprot [3] are prominent examples of public repositories extensively used by genomics researchers and practitioners, and biologists in general. Analysis and processing of low-quality data may result in wasted time and resources, or may lead scientists to false conclusions, thus hampering scientific progress.

Several quality models and assessment methodologies have been proposed in the literature, but most were developed in the context of enterprise data warehousing and addressed quality problems existing in the business domain. These methodologies do not fit naturally into the genomics context because biological data is more complex and less structured than typical business data. In addition, the increasing data generation and usage rates limit the kind of quality assessments that can realistically be performed. We therefore believe that there is a need for automated quality assessment techniques that provide users of genomics data sources with objective and quantitative estimates of the quality of available data.

* The author's current affiliation and address is: Microsoft Corp., One Microsoft Way, Redmond, WA 98052. Email: alexandm@microsoft.com

I. Măndoiu, R. Sunderraman, and A. Zelikovsky (Eds.): ISBRA 2008, LNBI 4983, pp. 469–480, 2008.

1.1 How Do Genomics Data Sources Currently Manage Quality?

To discover how public repositories of genomics data manage quality, we focused our study on the databases of the National Center for Biotechnology Information (NCBI) [4] because of their widespread use by the scientific community. The three major problems found related to quality are described next.

First, genomics data sources currently provide minimal information about the quality of the stored data. Some repositories offer base-calling scores, which are quality indicators of the sequence data solely. Typically, however, genomic records contain a significant amount of annotations about the sequence data, which should be accounted for if a comprehensive evaluation of the records is sought.

Second, the curation process that public genomics repositories have in place (which consists of cleaning, standardizing, and annotating the submitted data to improve its value and quality) is partially automated but still requires a significant amount of human effort. This, together with the increasing amount of data submitted by multiple sequencing centers on a daily basis, causes an increasing ratio of data generation to data curation. For this reason, most genomics sources publish their newly acquired data before it undergoes full curation, thus raising concern over the quality of available data.

Third, current query interfaces of genomics data sources do not support quality-driven queries. Without such capability, the identification of high-quality records from the query results becomes a time-consuming task for the users. While experienced users can generally glance at a record and roughly estimate its quality level, when a query retrieves a large number of records examining each record individually is not practical. New users would need to become familiar with the implicit quality indicators of the repository before they can properly interpret and use them. Hence, an automated way to present the query results ranked by quality score would be preferred.

1.2 Benefits of Quality Augmentation in Genomics Data Sources

Augmenting existing genomics repositories with quality information would have multiple benefits. First, the value and utility of existing repositories would be enhanced by providing users with quality information about the retrieved data, which would in turn help the users decide what data best fits their specific needs. Second, biologists and other users of current repositories would be able to work more effectively when using a quality-aware interface that allows them to filter out query results below a certain quality threshold, and to rank the retrieved data based on different quality scores. This would aid users to quickly discriminate high quality records from query results. Third, the data curation process would be facilitated by providing preliminary estimates for the quality of records submitted to the database, which can in turn help curators prioritize records for further revision.

2 Related Work

Numerous models, evaluation methodologies, and improvement techniques have been developed in the area of Information Quality (IQ). Particularly, Lee et al. developed

AIMQ [5], a methodology for IQ assessments and benchmarks based on a set of IQ dimensions important to information consumers. Naumann and Rolker [6] proposed an assessment-oriented classification of IQ criteria based on three sources of IQ: the user, the source, and the query process. These works represent valuable contributions for modeling and understanding data quality and its challenges, but they fail to provide quantitative measures that support the quality dimensions or indicators proposed.

Data Quality has also been studied in the context of Cooperative Information Systems. Mecella et al. [7] describe a service-based framework for managing data quality in cooperative information systems, based on an XML model for representing and exchanging data and data quality. Scannapieco et al. [8] developed the DaQuinCIS architecture and the Data and Data Quality model for managing data quality in cooperative information systems. Naumann et al. [9] presented a model for determining the completeness of a source or combination of sources. All of these works address quality issues that arise in the presence of multiple sources, in particular problems related to the exchange and integration of quality information among the sources. Most such issues, however, are not applicable to our model since we are primarily concerned with the quality of data within a single source.

A few works have been proposed in the context of data quality for biology and the life sciences. The research by Müller et al. [10] identifies the main errors involved in the process of genome data production as well as their corresponding data cleansing challenges. A thorough examination of the quality of the human genome DNA sequence is described by Schmutz et al. [11]. Both focus on assessing the quality of the sequenced data only, whereas our approach is also concerned with the annotations about the sequenced data. Missier et al. [12] proposed the Qurator system, which allows the specification of user's personal quality functions into quality views that are compiled into Web services. Preece et al. [13] describe a framework for managing information quality in e-Science, which allows scientists to define the quality characteristics that are of importance in their particular domain. Our approach differs from these works in that it aims to define general and objective quality dimensions that can be computed in an automated way, i.e., does not require user's input.

3 Quality Estimation Model

We present QEM, a new model for estimating the quality of biological data in genomics repositories. The model comprises a set of measurable quality dimensions, and a set of quantitative measures that can be systematically computed to provide a score for each quality dimension.

We define *Data Quality* as a measure of the value of the data. Since value is a rather intangible concept, we decompose it along five different quantifiable dimensions. *Quality dimensions* are aspects of the quality of data which either the user or the data provider is interested in measuring. Since we aim for quality dimensions that can be quantified, we need to specify how the quality dimensions will be measured. The particular formula or algorithm by which each dimension is assigned a score is called a *measure*. The set of quality scores (one per dimension) of a data item is referred as its *quality metadata*, and represented as a vector where each

entry contains the data item's score for a dimension, e.g., $Q = [d_1,d_2,\ldots, d_n]$ with $d_1,d_2,\ldots, d_n$ being the scores for the n quality dimensions.

3.1 Quality Dimensions and Measures

In order to identify suitable quality dimensions for our model, we looked for dimensions that met the following criteria. First, the dimension could be objectively measured, meaning that no subjective appraisal or interpretation was needed to assess a score for the dimension. Second, the measure for the dimension could be efficiently computed, meaning that initialization and update of the dimension's score was fast enough to allow its use in a real scenario. Third, the dimension was biologically relevant, meaning that it effectively captured criteria directly or indirectly used by biologists when assessing the quality of data. The biological relevance was initially judged by the authors and later validated experimentally.

Using these criteria, a set of seven quality dimensions was selected, namely *Density*, *Freshness*, *Age*, *Stability*, *Uncertainty*, *Linkage*, and *Redundancy*. The first five of these dimensions are *per*-record dimensions and the last two are *cross*-record dimensions. Per-record dimensions are dimensions that consider quality aspects of a single record (i.e., they assess records on an individual basis). Cross-record dimensions are dimensions that consider quality aspects across a set of records (i.e., they asses the interactions among records).

Underlying Data Model. Before formulating measures for the quality dimensions, we describe the data model in which the underlying biological data is represented: the semistructured data model. Semistructured data is commonly described as "schemaless" or "self-describing" data [14] because the schema is contained within the data. A semistructured data model generally represents data hierarchically (i.e., in a tree-like structure), with actual data represented at the leaf nodes and schema information encoded in upper layers of the tree (i.e., internal nodes). In this work, leaf nodes store atomic data items, which are either strings or numbers, and internal nodes represent complex data items, which are collections of data items. The Abstract Syntax Notation number One (ASN.1) [15] and the Extensible Markup Language (XML) [16] are two examples of semistructured data models.

Density Dimension and Measure. This spatial dimension assesses the amount of information conveyed by a data item d. The amount of information can be measured as the number of (possibly nested) data items within d. We include two density sub-dimensions that are relevant for biologists, *Features* and *Publications*. The former considers features annotated in the Feature Table [17] of a genomic record, which describe regions of biological significance in the sequence. The latter considers the references of a genomic record, i.e., publications in a journal article, book chapter, book, thesis, monograph, proceedings chapter, proceedings from a meeting, or patent.

The density score of an atomic data item d is defined as 1 for any d, hence each atomic data item has the same contribution to the total amount of information. Based on the density score of atomic data items, we recursively compute the density score of complex data items. If n is the number of components (i.e., direct descendants) of a complex data item d, and D_i is the density score of the ith component of d in the data tree, the density score of d is defined as

$$D = 1 + \sum_{i=1}^{n} D_i \ . \tag{1}$$

The density score can take on values from the interval [1, ∞[, where 1 represents the minimum density value, and there is no upper limit on the density value.

Freshness Dimension and Measure. This temporal dimension indicates how up to date the contents of a data item *d* are. It can be measured as a function of the time elapsed since the last update of the data item *d*, using an exponential decay. The freshness score of an atomic data item *d* is given by

$$F = e^{-\sigma\left(\frac{t-u}{f}\right)}, \tag{2}$$

where *t* is the current time, *u* is the time when d was last updated, *f* is the frequency of update of the database, and σ is a parameter that controls the decay rate of the freshness score. The role of *f* is to scale the time elapsed since last update to units that reflect the rate at which the database gets updated. The exponential decay gives more weight to recent past than to distant past, and also ensures that the freshness score takes values between 0 and 1.

For a complex data item *d*, the freshness score is defined as the average of the freshness scores of its components. The freshness score can take on values from the interval [0, 1], where 0 and 1 denote the minimum and maximum freshness values, respectively.

Age Dimension and Measure. This temporal dimension indicates how old the contents of a data item *d* are. It can be measured as a function of the time elapsed since the creation of the data item *d*. The age score of an atomic data item *d* is

$$A = 1 - e^{-\beta\left(\frac{t-c}{f}\right)}, \tag{3}$$

where *t* is the current time, *c* is the time when *d* was created, *f* is the frequency of update of the database, and β is a parameter that controls the decay rate of the age score. The role of *f* is to scale the time elapsed since creation to units that are in accordance to the database update rate. The transformation applied to this scaled time produces large increases in age at the beginning and then slows down as time passes by. This also ensures that the age score is between 0 and 1.

For a complex data item *d*, the age score is defined as the average over the age score of its components. The age score can take on values from the interval [0, 1], where 0 and 1 denote the minimum and maximum age values, respectively.

Stability Dimension and Measure. This temporal and provenance dimension captures information about changes in the contents of a data item *d* through time. Stability can be measured as the magnitude of the changes undergone by the data item relative to its size, and weighted by a function of the time elapsed since the change occurred. This weighting function diminishes the influence of older updates in favor of recent ones. The stability score of an atomic data item *d* is given by

$$S = 1 - \sum_{i=1}^{n} \Delta(d(i-1), d(i)) \int_{t_i}^{t_{i-1}} \lambda e^{-\lambda t} dt \ , \tag{4}$$

where n is the number of intervals at which we measure the stability of d, t_i is the time elapsed since the ith interval (with $t_0 \equiv \infty$), $d(i)$ is the state of d at interval i, and $\lambda > 0$ is a free parameter. The Δ function measures the fraction of d that changed between two consecutive intervals. The integral of the exponential function applies a time-decaying weight to the changes undergone by d, effectively giving more weight to recent changes than to old ones. We define $\Delta(d_1,d_2)$ for atomic data items d_1 and d_2 as

$$\Delta(d_1, d_2) = \begin{cases} \dfrac{\text{editDist}(d_1, d_2)}{\max\{\text{length}(d_1), \text{length}(d_2)\}} & \text{if } d_1, d_2 \text{ are strings} \\ \dfrac{|d_1, d_2|}{\max\{d_1, d_2\}} & \text{if } d_1, d_2 \text{ are numbers} \\ 1 & \text{otherwise} \end{cases} . \tag{5}$$

Note that $0 \le \Delta(d_1, d_2) \le 1$ for any pair (d_1, d_2). If d_1 and d_2 are numbers, Equation 7 assumes that they are positive. An approximation to the Edit Distance function can be used if efficiency is a major concern.

The equivalent incremental formulation for the stability measure is

$$S_{t_k} = 1 - I_{t_k} \ , \tag{6}$$

$$I_{t_k} = e^{-\lambda t_{k-1}} I_{t_{k-1}} + \left(1 - e^{-\lambda t_{k-1}}\right) \Delta\left(d(k-1), d(k)\right) , \tag{7}$$

with Equation 6 being an exponential moving average with memory depth $e^{-\lambda t_{k-1}}$.

The stability score of a complex data item d is defined as the average over the stability score of its components. The stability score can only assume values from the range [0,1], where 0 and 1 represents minimum and maximum stability, respectively.

Uncertainty Dimension and Measure. This spatial dimension is an indicator of the lack of evidence for the contents of a data item d, normally linked to the inherent imprecision of the experimental procedure used to obtain the data. The Uncertainty dimension can be measured as the fraction of ambiguous values contained in the data item. In genomics databases, ambiguous or uncertain values mainly come from the sequence data and are expressed as degenerate bases. The uncertainty score of an atomic data item d representing a sequence string is

$$U = \frac{\text{degenerateCount}(d)}{\text{length}(d)} \ , \tag{8}$$

where degenerateCount(d) is a function that counts the total number of degenerate bases in d, and length(d) is the size of the string represented by d (i.e., the total

number of bases in the sequence). Atomic values other than sequence strings do not have an uncertainty score. The uncertainty score can only take values in the range [0,1], where 0 and 1 represent minimum and maximum uncertainty, respectively.

Linkage Dimension and Measure. This spatial cross-record dimension provides information about the incoming and outgoing links of a data item *d* (a record, in this case). In genomics databases, records can be linked to other relevant records, published articles, etc. Such information is generally represented as an interaction graph consisting of a set of nodes that represent records and a set of directed edges (or links) between nodes that represent relationships between records.

We split the Linkage dimension into four mutually exclusive sub-dimensions, namely *Literature Links*, *Gene Links*, *Structure Links*, and *Other Links*. The Literature Links dimension comprises links to or from literature databases, specifically NCBI's PubMed, Online Mendelian Inheritance in Man (OMIM), and Online Mendelian Inheritance in Animals (OMIA). The Gene Links dimension accounts for links to or from gene and genome databases, in particular NCBI's Gene, HomoloGene, and Genomes. The Structure Links dimension contains links to or from structure and domain databases, specifically NCBI's Structure (MMDB), 3D Domains, and Conserved Domains (CDD). Lastly, the Other Links dimension covers all other links not included in any of the previous dimensions. This linkage division was devised with the help of expert collaborators at our university. All linkage sub-dimensions can be measured as a link count over the respective target databases. Each link contributes with one unit to the link count. In the NCBI's databases all links are two-way, so linkage scores effectively reflect both the number of outgoing and incoming links to/from record *r*. The score for each linkage dimension can take on values from the interval $[0, \infty[$, where 0 means that no link exists, and there is no upper limit on the value of the linkage score.

Redundancy Dimension and Measure. This spatial and cross-record dimension captures information about the number of redundant data items with respect to a given data item *d* (a record). In genomics databases, two records are considered redundant if their sequence similarity is significantly high. Annotations about the sequence data could also be incorporated into a general measure of redundancy, but this would lead to expensive string comparisons among records in the database, which turns impractical. Even measuring the redundancy at the sequence level would be computationally expensive if no extra information is provided by the data source. Luckily, our target NCBI databases run BLAST periodically over the stored records to pre-compute the "neighbors" (i.e., related sequences) of every record. Using this information, redundancy can be measured as the number of neighbors of a record.

The Redundancy score of a complex data item *d* representing record *r* is defined as the number of links from *r* to distinct entries in the same database as *r* (e.g., nucleotide-nucleotide or protein-protein relationships). This link count effectively counts the neighbors of *r*. The redundancy score can take on values from the interval $[0, \infty[$, where 0 means that no redundant data items exist, and there is no upper limit on the value of the redundancy score.

4 Quality Management Architecture

The Quality Management Architecture (QMA) enables the integration of the QEM with an existing genomics repository. The key design constraint for the QMA was to minimize the changes that needed to be made to the existing repository. Figure 1 shows our reference QMA. Its main components are the External Data Source (EDS) and the Quality Metadata Engine (QM-engine). The EDS is an existing genomics repository with limited access (e.g., administrators may impose restrictions in query time and frequency). The QM-engine is separately managed, and contains a cache of the EDS and a metadata source. At the core of the QM-engine is the Quality Layer, which handles data and metadata loading, maintenance, and querying.

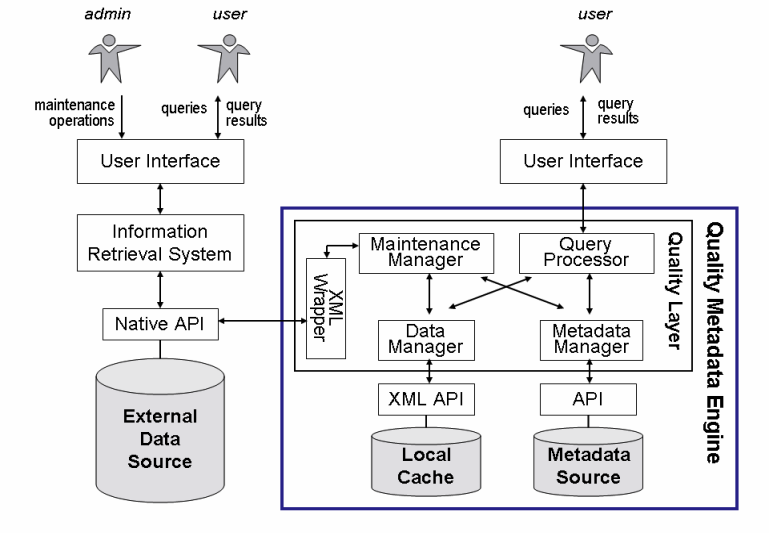


Fig. 1. Reference Quality Metadata Architecture

4.1 Implementation of the QMA and Operations

We implemented a QMA prototype system using the NCBI's Nucleotide and Protein databases as our EDS. Both the Entrez Programming Utilities [4] and the NCBI FTP site (ftp://ftp.ncbi.nih.gov/) were used to retrieve data from these databases. A modified version of the INSDSeq XML format (the official supported XML format of the International Nucleotide Sequence Database Collaboration [18]) was used to store data in the Local Cache. Oracle XML DB 10*g* [47] served as the DBMS for the Local Cache, while Oracle Database 10*g* [14] served as the DBMS for the Metadata Source. Further implementation details can be found in [19]. Next we outline the key macro operations that are part of the QMA: Bulk-loading, Maintenance, and Querying.

Bulk-loading takes an input data set from the EDS and loads it into the QM-engine. Bulk-loading involves the computation of all quality scores for the input data, the

storage of these quality scores in the local Metadata Source, and the replication of the biological data in the Local Cache.

Maintenance refers to the process of keeping the contents of the QMA updated with respect to changes in the EDS. Maintenance involves the detection of data changes in the EDS, the update of the Local Cache contents to enforce consistency with respect to the external database, and the update of the corresponding quality scores in the Metadata Source.

Querying refers to the processing of user requests to retrieve specific data and metadata. Basic query handling in the QMA involves queries that *i*) retrieve only data, *ii*) retrieve data plus quality metadata, and *iii*) retrieve a subset of the quality metadata (user indicates what dimensions to retrieve). Advanced queries would support user-specified filtering conditions on one or more quality scores, and sorting (ranking) of results based on a specified quality dimension.

5 Evaluation

The purpose of this evaluation was to determine the significance and usefulness of the chosen quality dimensions and measures in assessing the quality of genomic data. In particular, we tested the ability of our (individual and combined) quality estimates to discriminate high versus low quality data. A 2-point scale was used both to facilitate the quality assessment process for the domain experts and to prevent bias resulting from finer scales. The domain experts who participated in our research study [19] provided the data set used here, which consists of 187 "high quality" records (HQ set) and 184 "low quality" records (LQ set), selected from the NCBI's Nucleotide and Protein databases. The size of this data set was limited due to practical restrictions in both the number of experts we recruited for our study, and the number of records that each expert could reasonably evaluate. To address this limitation, further experiments were conducted over a larger data set sampled from different databases whose overall quality was rated by experts (results not shown here for space reasons but can be found in [19]).

The data set was loaded into the QMA prototype to obtain the quality metadata of each record. Then a logarithmic transformation was applied to the scores representing counts, and finally all scores were standardized. In the first experiment, we performed a Wilcoxon rank sum test (at the 5% significance level) over the standardized scores of the HQ and LQ sets for each quality dimension. Table 1 shows the results. In this table we can see that the test rejected the null hypothesis for five dimensions: Density, Features, Freshness, Publications, and Stability; which indicates that each of these dimensions is able to differentiate the HQ from the LQ set. Yet, considerable overlap between the distributions exists (boxplots not shown here for space reasons), making it difficult to find good classification thresholds based solely on one dimension.

In the second experiment, we built a decision tree classifier for the HQ and LQ sets using the joint quality scores of all dimensions. This classifier (built with C4.5 [20]) allowed us to find suitable thresholds (given by the decision rules), key predictive dimensions, and a single score combining all dimensions (given by the predicted posterior probabilities of each class). An analysis of the dimensions chosen as the top-most split-attributes across various classification trees (details in [19]) yielded the

following six key dimensions for classification: Uncertainty, Density, Age, Features, Literature-Links, and Publications. We observe that the Density, Features, and Publications dimensions were also found discriminating by our first experiment. The Uncertainty dimension also obtained a low *p*-value in the first experiment (although not small enough to reject the test), which confirms its relevancy for classification. We also note that at least one of the temporal dimensions (Age, Freshness, and Stability) was found relevant by either experiment. Differences in the sets of key/ discriminating dimensions obtained from our two experiments are expected since the first experiment considered how well each dimension differentiated the two quality classes given the entire data space, whereas the second experiment considered how well each dimension classified a small subset of the data space (since each node in the classification tree partitions the data space in subsequently smaller subsets).

Table 1. Results of the Wilcoxon rank sum test over standardized quality scores for the HQ and LQ sets. The medians of the HQ and LQ sets are shown as reference. The last two columns contain the test's *p*-value and outcome (i.e., null hypothesis H_0 rejected or not).

Dimension	HQ median	LQ median	P-value	H_0 rejected
Density	-0.171	-0.429	0.000	Yes
Features	-0.301	-0.321	0.000	Yes
Freshness	-0.407	-0.420	0.030	Yes
Publications	-0.205	-0.217	0.032	Yes
Stability	0.520	0.741	0.035	Yes
Uncertainty	-0.167	-0.167	0.052	No
Age	-0.430	0.128	0.100	No
Redundancy	-0.356	-0.356	0.476	No
Other Links	-0.326	-0.326	0.660	No
Gene Links	-0.276	-0.276	0.892	No
Literature Links	-0.236	-0.236	0.920	No
Structure Links	-0.273	-0.273	0.998	No

Table 2. Classifier performance when using all versus key-only dimensions

Dimensions	Generalization error (%)
All	31.3
Key	32.9

Table 2 shows the average generalization error (over a 10-fold cross-validation) of the classifier built using all and key-only dimensions. The average generalization error obtained when using all twelve dimensions (31.3%) and when using only the six key dimensions (32.9%) are comparable, which shows that the selected key dimensions are relevant for classifying high versus low quality data.

Although a detailed evaluation of the QMA is out of the scope of this work, we close this section by highlighting two of its most relevant capabilities: 1) the

non-intrusive augmentation of existing genomics repositories with quality metadata, which would enable a smooth transition from current to quality-aware data sources, and 2) the support for quality-aware queries, which would enable users to obtain the most valuable data for the task at hand.

6 Conclusions and Future Work

We developed a new model, QEM, for estimating the quality of data in genomics databases. Unlike previous related works, our model is based on quality dimensions that can be quantitatively measured using data already stored in the repositories. We also developed a quality management architecture, QMA, that enables the integration of QEM with existing genomics databases. The usefulness and biological significance of the QEM was evaluated using expert-feedback, gathered through a research study. Results of this evaluation show that it is possible to build a classifier, based on our quality estimates, that discriminates high from low quality data with a prediction accuracy of 69%. The usefulness of the QMA was highlighted based on its most relevant features.

This is the first work (to the best of our knowledge) that has addressed the use of metrics for understanding the quality of genomic datasets. We showed that the chosen metrics are useful; however, we would like to note that these metrics are preliminary in nature. Once a full-fledged QMA system is deployed and used widely, the set of metrics can be refined. Hence, improvements to the measures (e.g., with respect to efficiency) and/or extensions of the set of quality dimensions can be explored in the future. Another area for future work is the development of benchmarks for genomics data quality. We believe that more studies like the one we conducted among domain experts are needed so that future researchers benefit from larger and broader data sets.

References

1. Benson, D.A., Karsch-Mizrachi, I., Lipman, D.J., Ostell, J., Wheeler, D.L.: GenBank. Nucleic Acids Res. 35(Database issue), D21–D25 (2007)
2. Pruitt, K.D., Tatusova, T., Maglott, D.: NCBI reference sequences (RefSeq): A curated non-redundant sequence database of genomes, transcripts and proteins. Nucleic Acids Res. 35(Database issue), D61–D65 (2007)
3. Boeckmann, B., Bairoch, A., Apweiler, R., Blatter, M.C., Estreicher, A., Gasteiger, E., Martin, M.J., Michoud, K., O'Donovan, C., Phan, I., Pilbout, S., Schneider, M.: The SWISS-PROT protein knowledgebase and its supplement TrEMBL in 2003. Nucleic Acids Res. 31(1), 365–370 (2003)
4. Wheeler, D.L., Barret, T., Benson, D.A., Bryant, S.H., Canese, K., Chetvernin, V., Church, D.M., DiCuccio, M., Edgar, R., Federhen, S., Geer, L.Y., Kapustin, Y., Khovayko, O., Landsman, D., Lipman, D.J., Madden, T.L., Maglott, D.R., Ostell, J., Miller, V., Pruitt, K.D., Schuler, G.D., Sequeira, E., Sherry, S.T., Sirotkin, K., Souvorov, A., Starchenko, G., Tatusov, R.L., Tatusova, T.A., Wagner, L., Yaschenko, E.: Database resources of the National Center for Biotechnology Information. Nucleic Acids Res. 35(Database issue), D5–D12 (2007)

5. Lee, Y.W., Strong, D.M., Kahn, B.K., Wang, R.Y.: AIMQ: A Methodology for Information Quality Assessment. Information and Management 40(2), 133–146 (2002)
6. Naumann, F., Rolker, C.: Assessment Methods for Information Quality Criteria. In: Proceedings of the International Conference on Information Quality, pp. 148–162 (2000)
7. Mecella, M., Scannapieco, M., Virgillito, A., Baldoni, R., Catarci, T., Batini, C.: Managing Data Quality in Cooperative Information Systems. In: Spaccapietra, S., March, S., Aberer, K. (eds.) Journal on Data Semantics I. LNCS, vol. 2800, pp. 208–232. Springer, Heidelberg (2003)
8. Scannapieco, M., Virgillito, A., Marchetti, M., Mecella, M., Baldoni, R.: The DaQuinCIS Architecture: A Platform for Exchanging and Improving Data Quality in Cooperative Information Systems. Information Systems 29(7), 551–582 (2004)
9. Naumann, F., Freytag, J.C., Leser, U.: Completeness of integrated information sources. Information Systems 29(7), 583–615 (2004)
10. Müller, H., Naumann, F., Freytag, J.C.: Data Quality in Genome Databases. In: Proceedings of the International Conference on Information Quality, pp. 269–284 (2003)
11. Schmutz, J., Wheeler, J., Grimwood, J., Dickson, M., Yang, J., Caoile, C., Bajorek, E., Black, S., Chan, Y.M., Denys, M., Escobar, J., Flowers, D., Fotopulos, D., Garcia, C., Gomez, M., Gonzales, E., Haydu, L., Lopez, F., Ramirez, L., Retterer, J., Rodriguez, A., Rogers, S., Salazar, A., Tsai, M., Myers, R.M.: Quality assessment of the human genome sequence. Nature 429(6990), 365–368 (2004)
12. Missier, P., Embury, S., Greenwood, M., Preece, A., Jin, B.: Quality views: Capturing and exploiting the user perspective on data quality. In: Proceedings of the VLDB, pp. 977–988 (2006)
13. Preece, A.D., Jin, B., Pignotti, E., Missier, P., Embury, S.M., Stead, D., Brown, A.: Managing Information Quality in e-Science Using Semantic Web Technology. In: Sure, Y., Domingue, J. (eds.) ESWC 2006. LNCS, vol. 4011, pp. 472–486. Springer, Heidelberg (2006)
14. Abiteboul, S., Buneman, P., Suciu, D.: Data on the Web: From Relations to Semistructured Data and XML. Morgan Kaufmann Publishers, San Francisco, CA (2000)
15. Introduction to ASN.1, `http://asn1.elibel.tm.fr/en/introduction/`
16. Extensible Markup Language (XML) 1.0 (Second Edition), W3C Recommendation 6 (October 2000), `http://www.w3.org/TR/2000/REC-xml-20001006`
17. INSDC Feature Table Definition Document, http://www.insdc.org/files/feature_table.html
18. International Nucleotide Sequence Database Collaboration, `http://www.insdc.org/`
19. Martinez, A., Hammer, J.: BIODQ: A Model for Data Quality Estimation and Management in Biological Databases. Doctoral Thesis, University of Florida (2007)
20. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers, San Francisco, CA (1993)

Stepped Linear Regression to Accurately Assess Statistical Significance in Batch Confounded Differential Expression Analysis

Juntao Li[1], Jianhua Liu[2], and R. Krishna Murthy Karuturi[1,*]

[1] Computational and Mathematical Biology, [2] Systems Biology,
Genome Institute of Singapore, A*STAR (Agency for Science, Technology and Research), 60 Biopolis ST, S138672, Republic of Singapore
{lij9,liujh,karuturikm}@gis.a-star.edu.sg

Abstract. Batch effects in microarray experiments may lead to systematic shift in expression measurements from one batch to another. It poses great challenge if batches are confounded with the biological groups of interest especially in the estimation of statistical significance, FDR. Even the widely used well-tailored methods such as SAM are not immune to the effects of batch confounding of groups. We propose a *stepped linear regression (SLR)* method in the context of SAM to re-estimate the expected statistics and FDR in two class analysis to nullify batch effects and get really significant genes. SLR is equally applicable to the other similar methods and multi-group differential expression analysis.

Keywords: Differential expression, SAM, Microarray, Batch effect.

1 Introduction

Batch effects [7] are commonly observed across multiple batches of microarray experiments, There are many different kinds of effects, for example, RNA batch effect (experimenter, time of day, temperature), array effect (scanning level, pre/post-washing), location effect (chip, coverslip, washing), dye effect (dye, unequal mixing of mixtures, labeling, intensity), print pin effect, spot effect (amount of DNA in the spot printed on slide) [9] and or even the atmospheric ozone level [1]. Local batch effects (such as location, print pin, dye effect and spot effect) may be removed by using one of many local normalization methods available in the literature [6]. But global batch effects are too complicated and not easy to detect and eliminate in all circumstances.

The problem may be solved using linear model with batch as a factor or by empirical Bayes methods [5] if the experimental batches are not confounded with the biological groups i.e. each batch contains a mix of samples from different biological groups. Whereas the problem is not amenable to analysis if the biological groups are confounded with that of the batches i.e. the samples from one biological group belong to one batch and the other belongs to the other batch. The situation is unavoidable

* Corresponding author.

I. Măndoiu, R. Sunderraman, and A. Zelikovsky (Eds.): ISBRA 2008, LNBI 4983, pp. 481–491, 2008.

several times as one wants to compare the data from one experiment or lab to the data from another experiment or lab which essentially means batch confounded biological groups. The batch confounding is unavoidable in huge experiments even though all groups were generated in the same experiment. Batch confounding has severe influence on differential expression analysis as the biologically differentially expressed genes are mixed up with large number of mere batch affected genes. It may lead to underestimation of FDR (false discovery rate), to an intolerable limit, as several batch affected biologically irrelevant genes will also have significantly lower p-values. FDR is to be accurately estimated as it is an important parameter that complements the absence of *gold standard positive and negative test gene set* in genome-wide expression studies.

In this paper, we present and evaluate a method called *stepped linear regression (SLR)* to re-estimate the differential expression statistics in two class analysis under the assumption that the expression difference due to batch variation is smaller than that of the biological variation. And then we can adjust FDR based on the new expected statistics and get real biologically significant genes. We present our SLR in the context of SAM (Significance Analysis of Microarrays) [2]. SAM is a statistical technique for finding significantly differentially expressed genes in microarray experiments. SAM assigns a score to each gene on the basis of change in gene expression relative to the standard deviation of repeated measurements. For genes with scores greater than an adjustable threshold, SAM uses permutations of the repeated measurements to estimate the percentage of genes identified by chance, the false discovery rate (FDR). Using FDR, we can get the differentially expressed gene list by certain FDR threshold, those genes will be the significant genes according the data, but those are all not real biologically significant they may include several batch affected differentially expressed genes.

Though SLR is presented in the context of SAM two-group differential expression analysis for simplicity, the method is equally applicable to multiple-group studies affected by batch confounding and for any reasonable statistical procedure used. Our results show that SLR is effective in evaluating FDR accurately both in simulated as well as real data.

The remaining part of the paper is organized as follows: Section 2 to presents SLR in the context of SAM two-class analysis; section 3 presents evaluation of SLR on simulated and real data; and, section 4 presents discussion on SLR, the results and presents future directions.

2 Stepped Linear Regression (SLR)

The *stepped linear regression (SLR)* is based on the following assumptions: (1) there will be a systematic, but random, shift in expression measurements from one batch to another; (2) biological influence is greater than that of the batch effects' influence on biologically relevant genes; (3) the batch effect is independent of biological effect; and, (4) the proportion of biologically non-differentially expressed genes (π_0) is more than 0.5.

2.1 Re-estimate the Expected Statistics

In SAM two class analysis [3], we first get the SAM statistic d_i for each gene g_i (i = 1,2,..,n), and then get the order statistics $d_{(1)} \le d_{(2)} \le \ ... \le d_{(n)}$. To get FDR, we do permutations and get the expected order average statistics $\acute{d}_{(1)} \le \acute{d}_{(2)} \le \ ... \le \acute{d}_{(n)}$.

The linear model between SAM statistic $d_{(i)}$ and expected statistics $\acute{d}_{(i)}$ will be the following,

$$d_{(i)} = a * \acute{d}_{(i)} + b + c_i + e_i. \tag{1}$$

a and b are the batch effect factors, c_i is the biological effect factor and e_i is the error. If there are no batch effects, a will be 1 and b will be 0, and if gene g_i has no biological difference between two class experiments, c_i will be 0.

Then, we do the stepped linear regression to get the batch effect factors a and b as following procedure,

Step 1. *Do the linear regression for $d_{(i)}$ and $\acute{d}_{(i)}$ (i = 1,2,..,n) to get the slope a_1 and intercept b_1.*
Step 2. *Remove the gene g_k whose ($\acute{d}_{(k)}$, $d_{(i)}$) is maximally distant to the regression line and do linear regression for the rest of the genes and then get the new slope a_2 and new intercept b_2.*
Step 3. If $|a_2 - a_1| < \delta$ and $|b_2 - b_1| < \delta$, let $a = a_2$ and $b = b_2$; otherwise, let $a_1 = a_2$, $b_1 = b_2$ and repeat step 2.

After estimating the batch effect factors a and b, we can do the transformation for expected statistics as following,

$$\acute{d}^*_{(i)} = a * \acute{d}_{(i)} + b. \tag{2}$$

and then (1) will be

$$d_{(i)} = \acute{d}^*_{(i)} + c_i + e_i. \tag{3}$$

2.2 π_0 Estimate and FDR Adjustment

We assumed that the genes, which used to do linear regression at last to get the batch effect factor a and b, do not have biological difference, so we can set π_0, the proportion of true null (not biological effect) genes in the data set, as the proportion of those genes in the data.

The False Discovery Rate (FDR) is computed as ratio of median of the number of falsely called significant genes and the number of genes called significant.

3 Result

We show the effects of batch confounding on FDR estimation and the efficacy of SLR in alleviating it using both simulated data as well as real data. On simulated data, we show that SLR does not introduce artifacts in FDR estimation using data without batch effect confounding and show that it corrects the influence of batch effect confounding.

3.1 Simulation

A two-group data was simulated using the following rule

$$x_{ijk} = z_{ijk} + \eta_{ik} + \mu_{ik} \tag{4}$$

where x_{ijk} is an expression measurement of gene g_i (i = 1, 2…, N = 10000) in sample S_j (j=1, 2…10) in group G_k (k = 1, 2). $z_{ijk} \sim N(0, 1)$ is stand normal noise. The global batch effect η_{ik} and biological difference μ_{ik} are defined as follows:

$$\begin{gathered} \eta_{i1} = 0 \text{ for } 1 \le i \le N \\ \eta_{i2} = \theta_{i\eta} \sim N(0, \sigma_1^2) \text{ for } 1 \le i \le N \\ \\ \mu_{i1} = 0 \text{ for } 1 \le i \le N \\ \mu_{i2} = \theta_{i\mu} \sim N(0, \sigma_2^2) \text{ for } i \le n < N \\ \mu_{i2} = 0 \text{ for } n < i \le N . \end{gathered} \tag{5}$$

Where n is number of differentially expressed genes and N is the total number of genes. The model parameters signify that the batch effect and biological effect are different on different genes and the differential expression also varies from gene to gene. The fraction *{1-(n/N)}* is denoted by π_0, the fraction of non-differentially expressed genes or genes not affected by biological treatment. We simulated 4 different datasets of N=10000 genes using two different settings for each of σ_1 and π_0 as shown in table 1 while keeping σ_2=4.

Table 1. Parameters used to simulate the 4 different datasets A, B, C and D. δ = 0.0001.

Dataset	Dataset Simulation Parameters				New expected statistic ($d^*_{(i)}$)
	σ_1	σ_2	π_0 (n)	Batch Effect	
A	0	4	0.95 (500)	NO	1.10415 * $d_{(i)}$ -0.005924029
B	2	4	0.95 (500)	YES	3.889039 * $d_{(i)}$ -0.005366175
C	0	4	0.70 (3000)	NO	1.426328 * $d_{(i)}$ -0.004212276
D	2	4	0.70 (3000)	YES	4.148797 * $d_{(i)}$ -0.03025515

The datasets A and C are simulated without batch effects and analyzed with our procedure to show that our procedure does not introduce artifacts in the resultant FDR estimation i.e. difference between FDR estimates before correction and after correction should be close to zero for non-batch affected data. The datasets B and D are batch effect confounded data with reasonably different values of π_0 whose FDR estimates before correction are expected to be far from reality while the FDR estimates after adjustment are expected to be close to reality.

We used SAM on each of the four datasets to obtain d-statistics for original as well as permuted data. We applied our procedure on each pair of d-statistic sets with the model parameter δ to be 0.0001.

Table 2 shows the results for dataset *A* using SAM estimates and estimates after adjustment using our SLR procedure. The estimates of π_0 before (0.957) and after (0.954) adjustment are very close to each other and to the true value of 0.95. For different values of *delta* (a SAM parameter) threshold, table 2 shows the number of genes called significant and the respective FDR for original SAM and SAM followed by our procedure. The FDR estimates are very much close to each other while the number of genes called significant is different for *delta* close to 0 and 0.1 as the small change in FDR may result in huge change in number of genes called significant at very low values of delta.

Table 2. Significant gene table for dataset *A*

	before adjustment ($\pi_0 = 0.957$)		after adjustment ($\pi_0 = 0.9541$)	
delta	called	median FDR	called	median FDR
0.0	9026	0.9444358	5808	0.9426830
0.1	863	0.5211935	539	0.5168779
1.0	302	0.0000000	293	0.0000000
5.0	37	0.0000000	32	0.0000000
9.0	1	0.0000000	1	0.0000000

Table 3 shows the results of applying dataset *B*, similar to dataset *A* but with batch effect. The estimate of π_0 obtained from SAM (~0.264) is far below the true value of 0.95 while the same is much better estimated using our SLR adjustment procedure (~0.936) which close to the true value (0.95). Similar behavior is observed in the FDR estimates also for different values of *delta* thresholds as shown in table 3: the FDR estimates and number of genes called significant are very much different while the FDR before adjustment is unrealistically low (<0.27) at *delta*=0 and 0.1 whereas it is realistic (>0.65) after adjustment which can be mostly explained by the difference in estimates of π_0. Further, figure 1 shows the plots of estimated FDR and true FDR at each position of the gene ranking for both before and after SLR adjustment. Ideally, the curve should be diagonal from (0,0) to (1,1); however, closer the better. Both plots for dataset *A* are similar and close to diagonal for most part of the range of FDR except towards FDR=1 which is not so important anyway. Whereas, FDR plots for original SAM and SLR adjusted are quite different for dataset *B*, a batch affected data. Plot for SAM is almost close to x-axis shows how badly the FDR was

Table 3. Significant gene table for dataset *B*

	before adjustment ($\pi_0 = 0.2642$)		after adjustment ($\pi_0 = 0.9362$)	
delta	called	median FDR	called	median FDR
0.0	9988	0.2627451542	9995	0.93582533
0.1	9431	0.2152735553	1016	0.65331280
0.5	7019	0.0381488389	244	0.21678402
1	4434	0.0002383401	119	0.04720336
5	54	0.0000000000	2	0.00000000

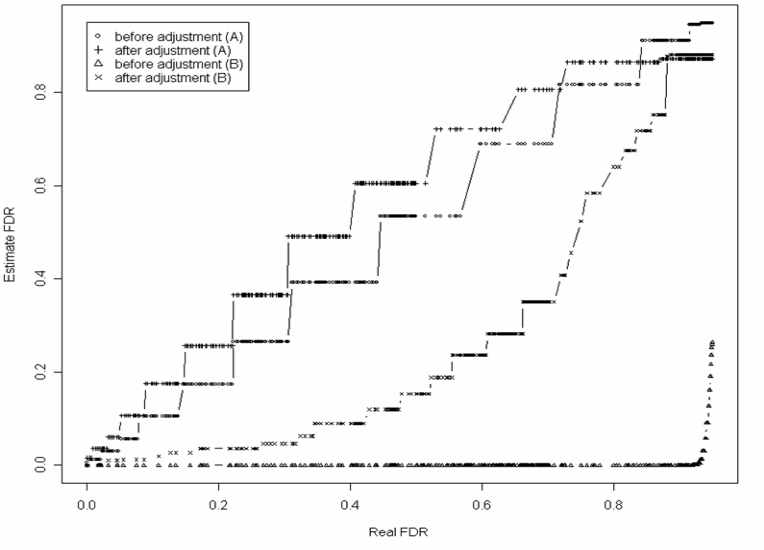


Fig. 1. FDR comparison for simulation data sets *A* and *B*. *Circle* and *plus* curves indicate FDR comparison before and after SLR adjustment respectively for data set *A*. They are, as required close to each other as well as close to the diagonal of the graph. It shows that blind application of SLR does not introduce any serious artifacts in to FDR estimation. Triangle and fork curves indicate FDR comparison before and after SLR adjustment respectively for data set *B*. FDR before SLR application is intolerably underestimated while the FDR after SLR application is, though underestimated, is much closer to the diagonal.

underestimated; whereas, FDR after SLR adjustment is much closer to diagonal than to x-axis, though still underestimated owing to the influence of batch effect on the permutation procedure [10].

Similar results are shown for datasets *C* and *D* in tables 4 and 5 respectively. The estimates of π_0 are consistently close to the true value of 0.7 irrespective of the presence of batch effect. But it is severely affected by batch effect in the original SAM application in the presence of batch effect. Similar differences exist even for FDR estimations for various values of delta. Figure 2 shows the plots of estimated FDR and true FDR at each position of the gene ranking for both before and after SLR

Table 4. Significant gene table for dataset *C*

	before adjustment ($\pi_0 = 0.7734$)		after adjustment ($\pi_0 = 0.6924$)	
delta	called	median FDR	called	median FDR
0.0	9865	0.77120485	7062	0.704165506
0.1	4774	0.58661110	2951	0.566402440
0.5	2356	0.02970828	2148	0.087678212
1	1932	0.00000000	1763	0.002749178
5	217	0.00000000	172	0.000000000

adjustment. Both plots for dataset *C* are similar and close to the diagonal for most part of the range of FDR except towards FDR=1 which is not critical in differential expression analysis. Whereas, FDR plots for original SAM and SLR adjusted are quite different for the batch affected dataset *D*. Plot for SAM is almost close to X-axis shows how bad the FDR estimation was; whereas, FDR after SLR adjustment is very much close to diagonal as required than to x-axis.

Table 5. Significant gene table for dataset *D*

	before adjustment ($\pi_0 = 0.2506$)		after adjustment ($\pi_0 = 0.7609$)	
delta	called	median FDR	called	median FDR
0.0	9992	0.2494839371	7050	0.7436313
0.1	9444	0.2096028590	2848	0.6130214
0.5	7464	0.0455773714	1440	0.2586532
1	5248	0.0003342607	829	0.0748050
5	374	0.0000000000	24	0.0000000

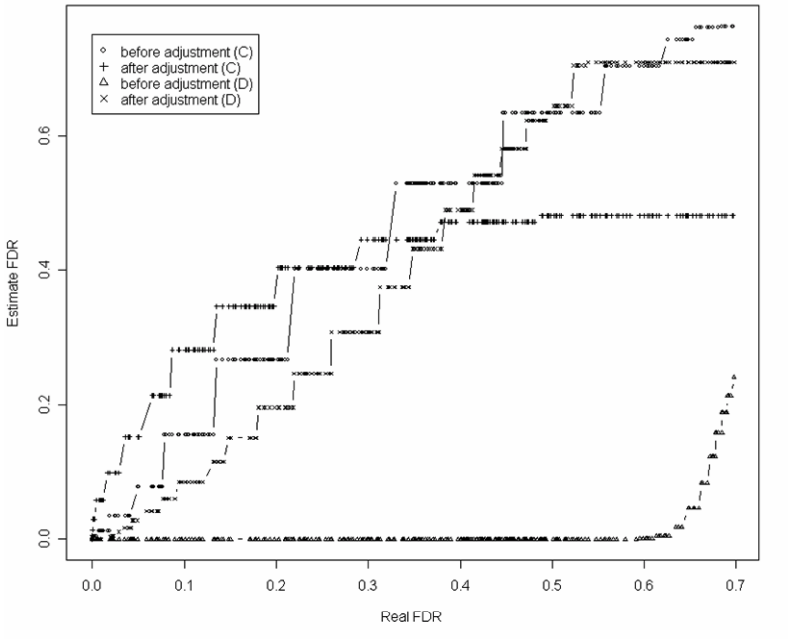


Fig. 2. FDR comparison for simulation data sets *C* and *D*. *Circle* and *plus* curves indicate FDR comparison before and after SLR adjustment respectively for data set *C*. They are, as required close to each other as well as close to the diagonal of the graph. It shows that blind application of SLR does not introduce any serious artifacts in to FDR estimation. Triangle and fork curves indicate FDR comparison before and after SLR adjustment respectively for data set *D*. FDR before SLR application is intolerably underestimated while the FDR after SLR application is, though underestimated, is much closer to the diagonal.

3.2 The *S. pombe* Data

At the next step, we show the utility of our approach on real gene expression data, *mip1* mutant (*Δmip1*) differential expression in *S. pombe* compared to its wild-type. The data was obtained from [4] containing 28 wt/wt spotted 2-color array data and 6 Δmip1/wt data with ~5000 *Open Reading Frames (ORFs)*. The purpose is to find the genes influenced by mip1 mutant (*Δmip1*) cells. The wt/wt data contains two batches of equal number of arrays which we call wt1/wt1 (or wt_rep1) and wt2/wt2 (or wt_rep2). The application of SAM on wt1/wt1 vs. wt2/wt2 data are shown in table 6 where it estimates π_0 to be 0.25 while it is supposed to be 1 as the both wt1 and wt2 samples are same except that they were hybridized onto two different batches of arrays at two different times. The corresponding SAM plot is shown in top-left of figure 3. After application of our SLR procedure, we obtained π_0 to be ~0.996 which is amazingly close to 1 as required and the respective SAM plot was shown in top-right of figure 3.

Table 6. Significant gene table for *S. pombe* wt1/wt1 vs. wt2/wt2. $d^*_{(i)} = 3.844653 * d_{(i)} - 0.2140583$, $\delta = 0.0001$.

	before adjustment ($\pi_0 = 0.2515723$)		after adjustment ($\pi_0 = 0.9959424$)	
delta	called	median FDR	called	median FDR
0.0	4486	0.16787378	3450	0.8731095
0.1	4414	0.15408378	3402	0.8661069
0.5	4151	0.10336223	533	0.5063797
1	3538	0.03075326	145	0.3022170
3	1100	0.00000000	10	0.0000000

We analyzed wt/wt (combine wt1/wt1 and wt2/wt2) vs. Δmip1/wt using SAM to identify the differentially expressed genes, the results are shown in table 7 and the corresponding SAM plot is shown in bottom-left of figure 3. Δmip1/wt was hybridized on altogether a different batch of arrays at completely different time resulted in batch effects again and the underestimation of π_0 (0.195) and FDR as shown in table 7 and bottom-left of figure 3. We applied our SLR procedure on this results as the corresponding estimates of π_0 and FDR are as shown in table 7 and bottom-right figure 3. π_0 estimate is more realistic (0.54) than the otherwise unrealistically estimated by SAM alone (0.195).

Table 7. Significant gene table for *S. pombe* wt/wt vs. Δmip1/wt. $d^*_{(i)} = 4.877984 * d_{(i)} + 0.01776157$, $\delta = 0.0001$.

	before adjustment ($\pi_0 = 0.1951714$)		after adjustment ($\pi_0 = 0.5491986$)	
delta	called	median FDR	called	median FDR
0.0	4866	0.1872298	4340	0.5516029
0.1	4757	0.1699185	2100	0.5626671
0.5	3961	0.05080074	1166	0.3699790
1	3074	0.00009523655	604	0.1463923
3	905	0.0000000000	138	0.0000000

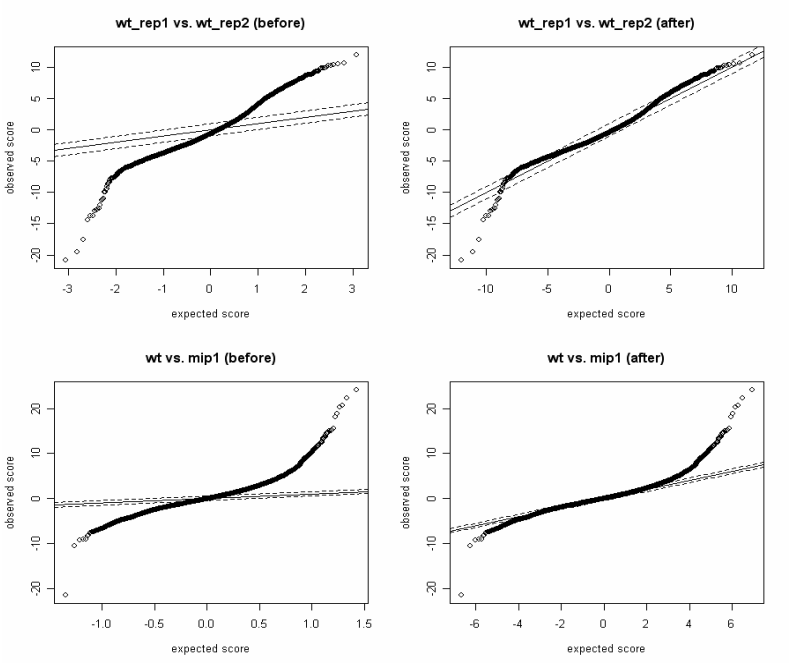


Fig. 3. SAM plot (before and after SLR adjustment) for *S. pombe* data set. Figure in top left (a) shows the SAM plot before SLR adjustment for wt1/wt1 vs. wt2/wt2 dataset, figure in top right (b) shows the SAM plot after adjustment for wt1/wt1 vs. wt2/wt2 dataset, figure in bottom left (c) shows the SAM plot before adjustment for wt/wt vs. Δmip1/wt dataset, figure in bottom right (d) shows the SAM plot after adjustment for wt/wt vs. Δmip1/wt dataset. The results are encouraging and SLR is a practically useful technique.

4 Discussion

We have proposed a *stepped linear regression* to correct for the FDR estimation artifacts introduced by batch confounding of treatment and control groups of samples. The problem is critical in several gene expression studies where one wants to compares the data obtained from different labs or from the same lab but at different times. Following several assumptions including the batch effects are small and influences all spots on the array in unexpected but definite manner in one batch but different in another batch. The influence is mainly on the estimation of FDR via badly underestimated proportion of non-differentially expressed genes and by the inevitable influence of change of mean value on permutation procedure, if adopted to estimate FDR. The SAM manual sites this behavior as one that could be biologically meaningful and the decision should be left to biologists. But, we feel that, in almost all gene expression studies π_0 is expected to be reasonably more than 0.5. Hence, we proposed SLR method, under the realistic assumptions of low batch effects, attempts to resolve this problem. SLR procedure is equally general for any differential expression analysis procedure for any number classes though it was described and evaluated in the

context of SAM, a popularly used method for differential expression analysis, for the sake of simplicity.

We have shown the efficacy of our SLR method on both simulated as well as real data. The results demonstrate that SLR combined with SAM is robust to batch confounding effects of treatments. They may be weakly demonstrating that SLR gives better estimate of π_0 than SAM alone in the absence of batch effects as shown in tables 2 and 4. To prove this point, we have to conduct an extensive set of experiments. Further, we feel that there is a lot more scope to improve SLR as shown in figure 1 that FDR after adjustment deviates from diagonal near FDR=1 for dataset *A* and it is considerably away from diagonal for dataset *B*. However the method in the current form is still useful in making right choice of differential expression threshold in the wake of better and meaningful FDR estimation.

Similar problem has been addressed in the evaluation of enrichment of gene sets in a list of genes [11], the GSA algorithm. GSA handles the problem by making the mean and standard deviations of the distributions of both observed statistics and permutation statistics. The idea is simple and effective for GSA as π_0 is close to 1. But it may not work as π_0 is reasonably less than 1 in several gene expression studies leading to severe overestimation of standard deviation making the idea ineffective for this purpose. Hence the utility of SLR plays an important role.

Acknowledgments

The authors would like to thank Haixia Li, Rehena Sultana, Vladimir Kuznetsov and Edison Liu for their valuable suggestions and support during this work. This work was supported by the Biomedical Research Council of A*STAR (Agency for Science, Technology and Research), Singapore.

References

1. Fare, T.L., Coffey, E.M., Dai, H., He, Y.D., Kessler, D.A., Kilian, K.A., Koch, J.E., LeProust, E., Marton, M.J., Meyer, M.R., Stoughton, R.B., Tokiwa, G.Y., Wang, Y.: Effects of atmospheric ozone on microarray data quality. Analytical Chemistry 75, 4672–4675 (2003)
2. Tusher, V.G., Tibshirani, R., Chu, G.: Significance analysis of microarrays applied to the ionizing radiation response. Proc. Natl. Acad. Sci. USA. 98, 5116–5121 (2001)
3. Chu G., Narasimhan B., Tibshirani R., Tusher V.G.: SAM, significance Analysis of Microarrays, Users guide and technical document
4. Chu, Z., Li., K.R.K.M., Lin, K., Liu, J.: Adaptive Expression Responses in the Pol-gamma Null Strain Depleted of Mitochondrial Genome in *S. pombe*. BMC Genomics 8, 323 (2007)
5. Johnson, W.E., Li, C., Rabinovic, A.: Adjusting batch effects in microarray expression data using empirical Bayes methods. Biostatistics 8(1), 118–127 (2007)
6. Smyth, G.K., Speed, T.: Normalization of cDNA microarray data. Methods 31(4), 265–273 (2003)
7. Lander, E.S.: Array of hope. Nature Genetics 21, 3–4 (1999)

8. Subramanian, A., Tamayo, P., Mootha, V.K., Mukherjee, S., Ebert, B.L., Gillette, M.A., Paulovich, A., Pomeroy, S.L., Golub, T.R., Lander, E.S., Mesirov, J.P.: A knowledge-based approach for interpreting genome-wide expression profiles. PNAS 102, 15545–15550 (2005)
9. Wit, E., McClure, J.: Statistical adjustment of signal censoring in gene expression experiments. Bioinformatics 19(9), 1055–1060 (2003)
10. Xie, Y., Pan, W., Khodursky, A.B.: A note on using permutation-based false discovery rate estimates to compare different analysis methods for microarray data. Bioinformatics 21(23), 4280–4288 (2005)
11. Efron, B., Tibshirani, R.: On testing the significance of sets of genes, Tech report. Stanford University (August 2006), http://www-stat.stanford.edu/~tibs/ftp/GSA.pdf

Bagging Multiple Comparisons from Microarray Data

Dimitris N. Politis

Department of Mathematics, University of California at San Diego,
La Jolla, CA 92093-0112, USA
dpolitis@ucsd.edu
http://www.math.ucsd.edu/∼politis

Abstract. Bagging and subagging procedures are put forth with the purpose of improving the discovery power in the context of large-scale simultaneous hypothesis testing. Bagging and subagging significantly improve discovery power at the cost of a small increase in false discovery rate with 'maximum contrast' subagging having an edge over bagging, i.e., yielding similar power but significantly smaller false discovery rates. The proposed procedures are implemented in a situation involving a well known dataset on gene expressions related to prostate cancer.

1 Introduction

The problem of simultaneous statistical inference is not new; see Miller (1981) for an early treatment. In the last decade, however, the statistical community has been faced with huge amounts of data and a subsequent need to address *large-scale* simultaneous hypothesis testing problems.

The prototypical such dataset involves gene expression data but different applications, such as functional Magnetic Resonance Imaging, flight spectroscopy, flow cytometry, etc., all give rise to similar problems from a statistician's perspective. The microarray set-up is described below in the context of the gene expression example with the understanding that the same ideas are applicable to a host of other two-sample, multiple comparison problems.

A typical experiment may entail data on n_X normal subjects, and n_Y patients. An array of N measurements is obtained from each subject. Therefore, the data can be organized as a $N \times n_X$ data matrix X (control group), and a $N \times n_Y$ data matrix Y (patient group); the (i, j) entry of X is denoted X_{ij}, and that of Y is denoted Y_{ij}. Column i from X has the data from the ith normal subject, and column j from Y has the data from the jth patient.

The X data are assumed independent of the Y data. A general model for this set-up is to assume that, for each k,

$$X_{k,1}, X_{k,2}, \ldots, X_{k,n_X} \sim \text{ i.i.d. } F_X^{(k)} \quad \text{and} \quad Y_{k,1}, Y_{k,2}, \ldots, Y_{k,n_Y} \sim \text{ i.i.d. } F_Y^{(k)} \quad (1)$$

where $F_X^{(k)}, F_Y^{(k)}$ are some distribution functions. For each $k = 1, \ldots, N$, the issue is to test $H_0 : F_X^{(k)} = F_Y^{(k)}$ vs. not; this is the set-up of multiple comparisons.

I. Măndoiu, R. Sunderraman, and A. Zelikovsky (Eds.): ISBRA 2008, LNBI 4983, pp. 492–503, 2008.

More often than not, the testing focuses on a potential difference in the means of the X and Y data. In that case, practitioners typically assume

$$X_{k,1}, X_{k,2}, \ldots, X_{k,n_X} \sim \text{ i.i.d. } N(\mu_k, \sigma_k^2) \tag{2}$$

and

$$Y_{k,1}, Y_{k,2}, \ldots, Y_{k,n_Y} \sim \text{ i.i.d. } N(\nu_k, \sigma_k^2). \tag{3}$$

The multiple comparisons now boil down to testing $H_0 : \mu_k = \nu_k$ vs. not, for $k = 1, \ldots, N$. From the kth row, the familiar t-statistic $t^{(k)} = (\bar{Y}_{k\cdot} - \bar{X}_{k\cdot})/(\hat{\sigma}\sqrt{n_Y^{-1} + n_X^{-1}})$ can be calculated where $\bar{Y}_{k\cdot} = n_Y^{-1}\sum_{j=1}^{n_Y} Y_{kj}$, $\bar{X}_{k\cdot} = n_X^{-1}\sum_{i=1}^{n_X} X_{ki}$, and $\hat{\sigma}^2 = (n_X + n_Y - 2)^{-1}\{\sum_{i=1}^{n_X}(X_{ki} - \bar{X}_{k\cdot})^2 + \sum_{j=1}^{n_Y}(Y_{kj} - \bar{Y}_{k\cdot})^2\}$ is the pooled variance.[1] A typical testing procedure then rejects H_0 from the kth row when $t^{(k)}$ is too large in absolute value.

Suppose that exactly n_0 rows (genes) conform to H_0, i.e., they are "null", and so $N - n_0$ rows (genes) do not, i.e., they are "non-null". Collect the indices of the truly non-null rows in a list denoted by TRUELIST; similarly, collect the row indices corresponding to the rejected t-statistics in the LIST of genes *declared* to be non-null. Then we can define the multiple comparisons *achieved discovery power* as

$$ADP = \frac{\#\{LIST \cap TRUELIST\}}{\#\{TRUELIST\}}$$

and the *achieved false discovery rate* as

$$AFDR = \frac{\#\{LIST \cap \overline{TRUELIST}\}}{\#\{LIST\}}$$

where $\#\{A\}$ denotes number of elements in set A, and $\bar{A}$ is the complement of A. The breakthrough method of Benjamini and Hochberg (1995) was designed to control the *expected value* of the AFDR; this expected value is usually called simply the false discovery rate (FDR).

2 Motivation

Suppose that two different groups perform the same scientific experiment and come up with two different lists of genes declared non-null, say $LIST_1$ and $LIST_2$. Let $AFDR_1$ and $AFDR_2$ denote the false discovery rates in the two experiments; recall that (the expected values of) $AFDR_1$ and $AFDR_2$ are controlled, i.e., bounded, in a typical multiple comparisons experiment.

How can the two lists, $LIST_1$ and $LIST_2$, be combined for better inference? The natural answer is to 'heed' the evidence from both experiments and declare

[1] The normality assumption is not crucial in practice, especially if the sample sizes n_X and n_Y are relatively large. Assuming common variance on the kth row of X and Y is more important; if in doubt, a slightly different form of the t-statistic must be used. In any case, the flavor of the testing problem remains unchanged.

as non-null all elements in the $BIGLIST = LIST_1 \cup LIST_2$. Since the BIGLIST is bigger than either $LIST_1$ or $LIST_2$, the combined experiment will have more power; but what is the AFDR associated with the BIGLIST?

To proceed with the analysis, let us make the simplifying assumption that genes declared non-null in both studies are very likely truly non-null, i.e., that $SMALLLIST \subset TRUELIST$ with high probability where $SMALLLIST \equiv LIST_1 \cap LIST_2$. Also let $FALSE_1$ denote the subset of $LIST_1$ that consists of false discoveries, i.e., genes falsely declared non-null; similarly for $FALSE_2$. Therefore, we have

$$AFDR_1 = \frac{\#\{FALSE_1\}}{\#\{LIST_1\}} \quad \text{and} \quad AFDR_2 = \frac{\#\{FALSE_2\}}{\#\{LIST_2\}} \tag{4}$$

from which the numbers $\#\{FALSE_1\}$ and $\#\{FALSE_2\}$ can be calculated as functions of $AFDR_1$ and $AFDR_2$.

Consequently, the AFDR associated with BIGLIST is given by:

$$\begin{aligned} AFDR_{BIG} &= \frac{\#\{FALSE_1\} + \#\{FALSE_2\}}{\#\{LIST_1\} + \#\{LIST_1\} - \#\{SMALLLIST\}} \\ &= \frac{AFDR_1 \times \#\{LIST_1\} + AFDR_2 \times \#\{LIST_2\}}{\#\{LIST_1\} + \#\{LIST_1\} - \#\{SMALLLIST\}}. \end{aligned} \tag{5}$$

Taking expectations in the above[2] we see that eq. (5) is satisfied with the expected false discovery rates (FDR) in place of the AFDRs, i.e., that:

$$FDR_{BIG} = \frac{FDR_1 \times \#\{LIST_1\} + FDR_2 \times \#\{LIST_2\}}{\#\{LIST_1\} + \#\{LIST_1\} - \#\{SMALLLIST\}}. \tag{6}$$

In experiments with low power it is not uncommon to have $LIST_1$ and $LIST_2$ be totally disjoint; see Efron (2006) for a discussion. Suppose we are in such a low-power set-up, and also suppose—for the sake of argument—that the two experiments have similar design, i.e., that $FDR_1 = FDR_2$. Then, the above equations show that $FDR_{BIG} = FDR_1 = FDR_2$. So, in this case, the combined experiment has more power with the *same* FDR, i.e., a win-win situation.

In general, however, $LIST_1$ and $LIST_2$ might not be disjoint, and the increase in power associated with BIGLIST will come at the price of an increase in FDR. However, it is the thesis of this paper that the increase in power may be well worth a small increase in FDR.

Before proceeding further, let us momentarily consider the generalization to the case of having M different groups perform the same experiment and coming up with their respective non-null lists, say $LIST_1, LIST_2, \ldots, LIST_M$; let $AFDR_1, AFDR_2 \ldots, AFDR_M$ denote the respective AFDRs. Under the same simplifying assumption, namely that genes declared non-null in at least two studies are very likely truly non-null, a similar calculation as before yields:

[2] Strictly speaking, this is a conditional expectation treating the size variables $\#\{LIST_1\}$, $\#\{LIST_2\}$ and $\#\{SMALLLIST\}$ as given.

$$FDR_{BIG} = \frac{\sum_{i=1}^{M} FDR_i \times \#\{LIST_i\}}{\#\{BIGLIST\}} \tag{7}$$

where FDR_{BIG} is the expected false discovery rate associated with $BIGLIST = \cup_{i=1}^{M} LIST_i$. Finally, note that the number of elements in $BIGLIST$ can be calculated as: $\#\{BIGLIST\} = \sum_i \#\{LIST_i\} - \sum_{i \neq j} \#\{LIST_i \cap LIST_j\}$

$$+ \sum_{i \neq j \neq k \neq i} \#\{LIST_i \cap LIST_j \cap LIST_k\} + \ldots + (-1)^{M-1} \times \#\{\cap_{i=1}^{M} LIST_i\}.$$

3 Bootstrap and Bagging

In Section 2, having multiple experiments (with their associated rejection LISTs) was discussed. In practice, however, the statistician is faced with a single dataset. Nonetheless, *resampling* and *subsampling* methods can be utilised in order to create additional (pseudo)samples.

Efron's (1979) *bootstrap* is one of the most prominent resampling methods. For i.i.d. data $Z_1, \ldots, Z_n$, the bootstrap amounts to sampling randomly with replacement from the set $\{Z_1, \ldots, Z_n\}$ to create the (pseudo)sample $Z_1^*, \ldots, Z_n^*$; see Efron and Tibshirani (1993) for a review. The bootstrap is closely related to Tukey's (1958) 'delete-1' *jackknife* which was generalized to a 'delete-d' jackknife by Shao and Wu (1989). For i.i.d. data $Z_1, \ldots, Z_n$, the delete-d jackknife is equivalent to *subsampling* with sample size $b = n - d$, i.e., sampling randomly *without* replacement from the set $\{Z_1, \ldots, Z_n\}$ to create the (pseudo)sample $Z_1^*, \ldots, Z_b^*$; see Politis, Romano and Wolf (1999).

'Bagging', i.e., **b**ootstrap **agg**regation, was put forth by Breiman (1996) in order to improve the accuracy of statistical predictors. The idea is to evaluate the predictor in question on a number of bootstrap (pseudo)datasets, and to combine the resulting predictors in an aggregate predictor. It has been shown that bagging indeed helps improve predictor accuracy in particular when the predictor is relatively unstable, i.e., when small changes in the data result in greatly perturbed predictions; see Bühlmann and Yu (2002). Bagging can alternatively be implemented in conjunction with subsampling in which case it is termed *'subagging'*; see Bühlmann and Yu (2002), and Bühlmann (2003).

4 Balanced Bagging and Subagging for Microarrays

As discussed in Section 2, it is possible to have two different low-power experiments produce disjoint or almost disjoint rejection lists; this is evidence of *instability*. Thus, bagging and/or subagging may be helpful for multiple comparisons as they have been shown to be helpful in prediction and classification.

We now elaborate on how to perform bagging and subagging in the multiple comparisons, microarray set-up of Section 1; the main idea is to re/sub-sample subjects, i.e., columns of the matrices X and Y. Throughout this section it is assumed that the practitioner is using a fixed multiple hypothesis testing

procedure, e.g., the procedure of Benjamini and Hochberg (1995) or Efron (2005), for *any* dataset that he/she may encounter.

The bagging and subagging algorithms described below are termed 'balanced'; the reason for this term will become more apparent in Section 6. Let $\underline{x}_1, \ldots, \underline{x}_{n_X}$ and $\underline{y}_1, \ldots, \underline{y}_{n_Y}$ denote the columns of X and Y respectively; B is an integer denoting the number of (pseudo)samples generated.

- **Balanced Bagging.** For $k = 1, \ldots, B$, construct the kth bootstrap (pseudo) sample $X^{(k)}$ and $Y^{(k)}$; the columns of $X^{(k)}$ and $Y^{(k)}$ respectively are given as $\underline{x}_{I_1}, \ldots, \underline{x}_{I_{n_X}}$ and $\underline{y}_{J_1}, \ldots, \underline{y}_{J_{n_Y}}$ where $I_1, \ldots, I_{n_X}$ are numbers drawn randomly with replacement from the index set $\{1, \ldots, n_X\}$ and $J_1, \ldots, J_{n_Y}$ are numbers drawn randomly with replacement from the index set $\{1, \ldots, n_Y\}$ and independently of $I_1, \ldots, I_{n_X}$. From this kth (pseudo)sample, the rejection list $LIST_k$ is created.

To define subagging, subsample sizes b_X and b_Y must be specified. Note that there is no reason here to have the subsample sizes be of smaller order of magnitude as compared to the original sample sizes; this is only required for estimation consistency which is not the objective here—see e.g. Politis et al. (1999). So, the subsample sizes for subagging could (and should) be taken relatively large; furthermore, it is intuitive that a choice satisfying $b_X/b_Y \simeq n_X/n_Y$ might be fruitful as being more representative of the original dataset. Thus, a good rule-of-thumb may be to let $b_X \simeq a\ n_X$ and $b_Y \simeq a\ n_Y$ where the constant a is close to (but less than) one.

- **Balanced Subagging—Random Version.** For $k = 1, \ldots, B$, construct the kth subagging (pseudo)sample $X^{(k)}$ and $Y^{(k)}$; the columns of $X^{(k)}$ and $Y^{(k)}$ respectively are given as $\underline{x}_{I_1}, \ldots, \underline{x}_{I_{b_X}}$ and $\underline{y}_{J_1}, \ldots, \underline{y}_{J_{b_Y}}$ where $I_1, \ldots, I_{b_X}$ are numbers drawn randomly *without* replacement from the index set $\{1, \ldots, n_X\}$ and $J_1, \ldots, J_{b_Y}$ are numbers drawn randomly *without* replacement from the index set $\{1, \ldots, n_Y\}$ and independently of $I_1, \ldots, I_{b_X}$. As before, from this kth (pseudo)sample, the rejection list $LIST_k$ is created.
- **Balanced Subagging—Nonrandom Version.** Let $\mathcal{S}_X$ denote the set of all size b_X subsets of the index set $\{1, \ldots, n_X\}$, and $\mathcal{S}_Y$ denote the set of all size b_Y subsets of the index set $\{1, \ldots, n_Y\}$ where b_X and b_Y are as above. A subagging (pseudo)sample is given by $X^{(k_1)}$ and $Y^{(k_2)}$ where the columns of $X^{(k_1)}$ are the columns of X with indices given by the k_1th element of set $\mathcal{S}_X$, and the columns of $Y^{(k_2)}$ are the columns of Y with indices given by the k_2th element of set $\mathcal{S}_X$. Since the set $\mathcal{S}_X$ contains $\binom{n_X}{b_X}$ elements and the set $\mathcal{S}_Y$ contains $\binom{n_Y}{b_Y}$ elements, it is apparent that there are $B = \binom{n_X}{b_X} \cdot \binom{n_Y}{b_Y}$ possible (pseudo)samples.

Of course, $\binom{n_X}{b_X} \cdot \binom{n_Y}{b_Y}$ can be a prohibitively large number, so considering *all* possible (pseudo)samples seems out of the question. The aforementioned random subagging procedure side-steps this difficulty but so does the following scheme that has the additional benefit of nonrandom selection of 'maximum

contrast' subsamples, i.e., subsamples that are 'most' different from one another in their composition. 'Maximum contrast' subsampling is somewhat reminiscent of subsampling for time series when partial block-overlap is used; see Politis et al. (1999, Ch. 9.2) and the references therein. To describe this notion we may equivalently use the 'delete-d' framework (with $d = n - b$) as opposed to 'choose-b'; of course, now the game is delete-d *columns* from one of our data matrices.

– **'Maximum Contrast' Nonrandom Subagging.** Let m_X, m_Y be two positive integers, and divide the index set $\{1, \ldots, n_X\}$ into the m'_X subsets $S_X^{(1)}, \ldots, S_X^{(m'_X)}$ where $S_X^{(1)} = \{1, \ldots, d_X\}, S_X^{(2)} = \{d_X + 1, \ldots, 2d_X\}, \ldots$, etc. where $d_X = \lceil n_X/m_X \rceil$ and $m'_X = \lceil n_X/d_X \rceil$; here $\lceil a \rceil$ is the smallest integer that is bigger or equal to a. The last set, i.e., $S_X^{(m'_X)}$, may have size less than d_X if m_X does not divide n_X but that poses no problem. Similarly, divide the index set $\{1, \ldots, n_Y\}$ into the m'_Y subsets $S_Y^{(1)}, \ldots, S_Y^{(m'_Y)}$ where $S_Y^{(1)} = \{1, \ldots, d_Y\}, S_Y^{(2)} = \{d_Y + 1, \ldots, 2d_Y\}, \ldots$, etc. A subagging (pseudo) sample is now given by $X^{(k_1)}$ and $Y^{(k_2)}$ where the columns of $X^{(k_1)}$ are the columns of X with indices given by the set $\{1, \ldots, n_X\} - S_X^{(k_1)}$, and the columns of $Y^{(k_2)}$ are the columns of Y with indices given by the set $\{1, \ldots, n_Y\} - S_Y^{(k_2)}$. Since the possible values of k_1 are $\{1, \ldots, m'_X\}$, and those for k_2 are $\{1, \ldots, m'_Y\}$, it is apparent that there are $m'_X \cdot m'_Y$ possible such (pseudo)samples; thus rejection lists $LIST_1, \ldots, LIST_B$ can be created with $B = m'_X \cdot m'_Y$.

5 Combining the Rejection Lists

Let $LIST$ denote the rejection list of the original dataset X and Y, and $LIST_1$, $\ldots, LIST_B$ the rejections lists corresponding to B (pseudo)samples from one of the algorithms of Section 4. As in Section 2, the simplest suggestion is to combine the lists by a union, i.e., to define the aggregate/combined list as:

$$LIST.AGG = LIST \cup LIST1 \cup LIST2 \cup \cdots \cup LISTB. \tag{8}$$

However, other alternatives exist; their description is facilitated by the notion of 'voting' where a list is said to 'vote' that the ith gene is non-null when the ith gene is an element of the list.

Let $V(i)$ denote the number of votes the ith gene received from the 'voting' lists $LIST$, $LIST_1, \ldots, LIST_B$. With this terminology, rejecting every gene in $LIST.AGG$ corresponds to the formula:

(i) declare the ith gene as non-null if $V(i) \geq 1$, i.e., it got at least one vote.

A more conservative approach might require to 'second' a vote, i.e., it would

(ii) declare the ith gene as non-null if $V(i) \geq 2$, i.e., it got at least two votes.

One might even raise the rejection threshold at a level higher than two although we will not consider that here. However, it is informative to see which genes

received more votes than others in the sense that getting more votes corresponds to more evidence for being truly non-null. Thus, a plot of $V(i)$ vs. i may be a helpful diagnostic tool.

As a further diagnostic, we may define $N(h)$ as the number of genes that received at least h votes, i.e., $N(h)$ is the size of the non-null list obtained from a criterion of the type: reject gene i if $V(i) \geq h$. A plot of $N(h)$ vs. h is another way to quantify the 'strength of evidence' towards proclaiming each gene on $LIST.AGG$ as non-null.

Note that formula (ii) treats $LIST$ as 'equal' to $LIST_1, \ldots, LIST_B$, and carries the implicit risk that not all of the genes found in $LIST$ will be finally rejected. To remedy this, we may give the original $LIST$ more weight in the aggregation. The easiest way of doing this is giving the original $LIST$ a double vote, i.e., defining $V^*(i)$ to equal the number of votes the ith gene got from $LIST_1, \ldots, LIST_B$ plus a double vote from the original $LIST$ (if indeed $LIST$ gave it a vote), and then

(ii) declaring the ith gene as non-null if* $V^*(i) \geq 2$.

As above, we can define $N^*(h)$ as the number of genes that received at least h votes from formula (ii*) above, i.e., $N^*(h)$ is the size of the non-null list obtained from a criterion of the type: reject gene i if $V^*(i) \geq h$. A plot of $N^*(h)$ vs. h has an interpretation similar to that of plot of $N(h)$ vs. h.

6 Comparison to Bagging for Classification

Microarray data, such as the ones arising in gene expression data, lend themselves to analysis with the objective of classifying future observations; in other words, using the data to decide if a future observation belongs to the control or the patient group—the decision being based on the new observation's 'features' (i.e. gene expressions) only. Since Breiman's (1996) original bagging was aimed at improving predictors and classifiers, it is of no surprise that there is already a body of literature on bagging and subagging microarrays with the purpose of classification; a partial list includes Dettling (2004), Dudoit and Fridlyand (2003), and Dudoit, Fridlyand and Speed (2002).

Although related at the outset, classification is a very different problem than hypothesis testing; their objectives are quite different, and so are the methods involved. To illustrate this point, we now give a brief description of the bagging/subagging procedures as used for microarray classification.

To start with, concatenate the X and Y matrices into a big $N \times n$ matrix denoted by W where $n = n_X + n_Y$. Let $\underline{w}_1, \ldots, \underline{w}_n$ denote the columns of W, and define new variables $U_1, \ldots, U_n$ such that $U_i = 0$ for $i \leq n_X$, and $U_i = 1$ for $i > n_X$; in this sense, the variable U_i is an indicator of which group (normal or patient) the ith subject belongs to. Finally, define $Z_i = (\underline{w}_i, U_i)$ for $i = 1, \ldots, n$. The Z_i data are multivariate but they constitute a *single* sample that can be bootstrapped—by sampling with replacement from the set $\{Z_1, \ldots, Z_n\}$, or subsampled—by sampling without replacement from the same set $\{Z_1, \ldots, Z_n\}$,

in order to create (pseudo)samples. In all the above-referenced works, bagging/subagging for microarray classification follows the above paradigm.

Note, however, that the above single-sample bootstrap scheme can generate (pseudo)samples that are *unbalanced* in terms of the two groups (normal/patient). To elaborate, let $Z_i^* = (\underline{w}_i^*, U_i^*)$ for $i = 1, \ldots, n$ be the bootstrap (pseudo)sample. Then, it is not unlikely that $\sum_{i=1}^n U_i^*$ turns out quite different from its expected value of n_Y; in fact, it is even possible (although very unlikely) that $\sum_{i=1}^n U_i^*$ is 0 or n, i.e., the (pseudo)sample consisting of data from one group only.

The above discussion refers to bootstrap and bagging but similar ideas hold for single-sample subagging. Let us define a (pseudo)sample to be *balanced* if the proportion of patients to control subjects within the (pseudo) sample is equal to that found in the original sample, i.e., n_Y/n_X. If we let $Z_i^* = (\underline{w}_i^*, U_i^*)$ for $i = 1, \ldots, b$ be the subsampling (pseudo)sample, then it is still possible to have $\sum_{i=1}^n U_i^* = 0$ provided of course that $b \leq n_X$. But even barring such extreme events, it is clear that there is no guarantee that the above subsampling (pseudo)sample would be balanced.

In conclusion, the possibility of unbalanced (pseudo)samples might not adversely influence the properties of bagging/subagging for classification purposes but it is problematic in our hypothesis testing setting. The balanced bagging and subagging procedures of Section 4 are devoid of this deficiency, since they yield—by design—exactly balanced (pseudo)samples.

Finally, note that different resampling methods have been used in connection with multiple comparisons—the most popular of which involving permutation tests; see e.g. Westfall and Young (1993), Ge, Dudoit and Speed (2003), or Romano and Wolf (2004). In addition, the re-calculation of rejection lists over subsamples was considered by Newton et al. (2004) for the purpose of validating the stability of a particular list-forming method. Nevertheless, the approach of Section 4 constitutes the first—to our knowledge—application of the notion of bagging/subagging for the purpose of increasing detection power in multiple comparisons.

7 Some Concluding Remarks and a Real Data Example

In this paper, bagging and subagging procedures are proposed for improving an experiment's discovery power at the cost of a somewhat increased FDR. If it is required to exactly control the FDR of the bagged/subagged experiment to a certain level α, say, then the target FDR of each (pseudo)sample experiment must be chosen to be less than α; this choice of a smaller FDR for the (pseudo)sample experiments could be found as a result of a *calibration* procedure for which simulation experiments are helpful. An example of such a simulation is given in: `www.math.ucsd.edu/~politis/PAPER/Bagging.pdf` where, in addition, bagging is compared to subagging for efficacy. As intuited in the discussion of Section 4, 'maximum contrast' subagging is found to generally have an edge over bagging, yielding similar power increases but significantly less FDR.

Table 1. Numbers of non-null genes as found by applying Efron's `locfdr` method (first column), and 'maximum contrast' subagging using `locfdr` on each (pseudo)sample; 'thr' indicates the `locfdr` threshold

	data	subag (i)	subag (ii*)
thr=0.2	34	113	101
thr=0.3	51	142	123

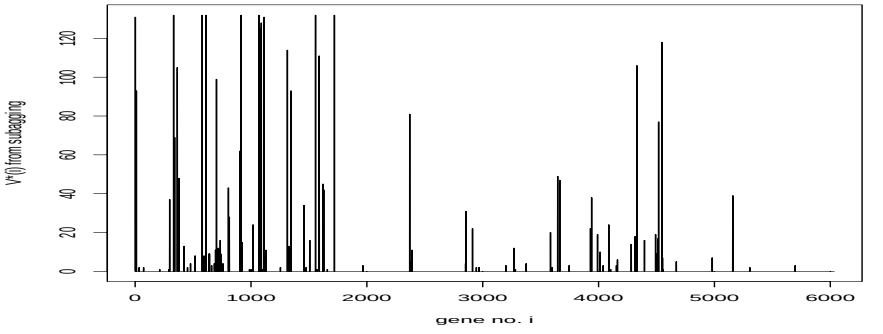


Fig. 1. Plot of function $V^*(i)$ vs. i in subagging the prostate data

To conclude, we now apply subagging to the well-known prostate cancer dataset of Singh et al. (2002) that has been analyzed extensively by Efron (2006); this is a 'low power' experiment, and thus could potentially benefit most from subagging. In the prostate dataset, there are $n_X = 50$ normal subjects, and $n_Y = 52$ patients; on each subject expression levels for $N = 6033$ are recorded.

To apply 'maximum contrast' subagging, the simple choices $m_X = 10$ and $m_Y = 13$ were used mostly for divisibility purposes; they correspond to delete-d with $d_X = 5$ and $d_Y = 4$. The data were pre-processed via a cube-root trasnformation as in, for example, Tusher, Tibshirani and Chu (2001). Efron's (2005) `locfdr` method was used to perform the multiple comparisons using two different thresholds, thr=0.2 and thr=0.3. The rejection lists for the original data, and formula (i) and (ii*) 'maximum contrast' subagging were compiled and given in the Appendix; their sizes are given in Table 1 where it is seen that subagging roughly *triples* the number of genes declared non-null.

Because of the potential increase in FDR that comes with bagging, the lower threshold thr=0.2 might be recommended—which is also `locfdr`'s default. Of the subagging formulas, one might prefer formula (ii*) subagging for reasons of being conservative. The plot of function $V^*(i)$ corresponding to the default threshold is given in Figure 1 where it is apparent that there are many genes that got an enormous number of votes; in fact, there are seven genes that were voted by the original list as well as *every* subagging list.

This phenomenon is shown clearly in the plot of function $N^*(h)$ of Figure 2. The left hand side end of the plot (where h equals one or two) corresponds to

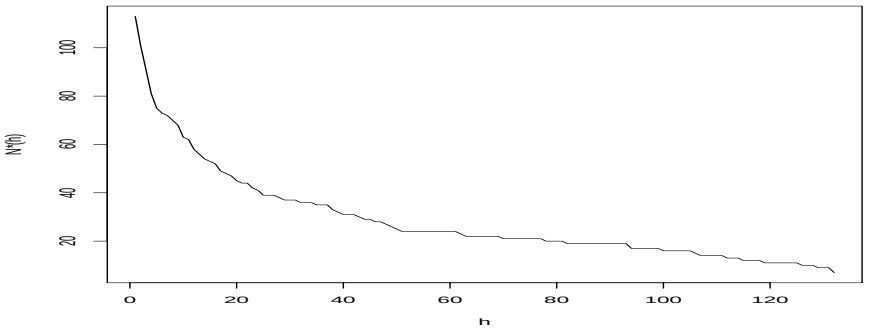


Fig. 2. Plot of function $N^*(h)$ vs. h in subagging the prostate data

the respective sizes (113 and 101) of the formula (i) and (ii*) lists mentioned above. The right hand side end of the plot corresponds to the case $N^*(132) = 7$, i.e., the seven genes voted by every list.

Acknowledgement. The idea for this paper was inspired by Brad Efron's talk "Doing Thousands of Hypothesis Tests at the Same Time"; the author is grateful to Prof. Efron for many helpful discussions, and for sharing his software and data. The support of NSF via grant SES-04-18136 is also gratefully acknowledged.

References

1. Benjamini, Y., Hochberg, Y.: Controlling the false discovery rate: A practical and powerful approach to multiple testing. J. Roy. Statist. Soc., Ser. B 57, 289–300 (1995)
2. Breiman, L.: Bagging predictors. Machine Learning 24, 123–140 (1996)
3. Bühlmann, P.: Bagging, subagging and bragging for improving some prediction algorithms. In: Akritas, M.G., Politis, D.N. (eds.) Recent Advances and Trends in Nonparametric Statistics, North Holland, pp. 19–34. Elsevier, Amsterdam (2003)
4. Bühlmann, P., Yu, B.: Analyzing bagging. Ann. Statist. 30, 927–961 (2002)
5. Dettling, M.: BagBoosting for tumor classification with gene expression data. Bioinformatics 20(18), 3583–3593 (2004)
6. Dudoit, S., Fridlyand, J.: Bagging to improve the accuracy of a clustering procedure. Bioinformatics 19(9), 1090–1099 (2003)
7. Dudoit, S., Fridlyand, J., Speed, T.: Comparison of discrimination methods for the classification of tumors using gene expression data. J. Amer. Statist. Assoc. 97(457), 77–87 (2002)
8. Efron, B.: Bootstrap methods: Another look at the jackknife. Ann. Statist. 7, 1–26 (1979)
9. Efron, B.: Local false discovery rates (2005), `http://www-stat.stanford.edu/~brad/papers`
10. Efron, B.: Size, power, and false discovery rates (2006), `http://www-stat.stanford.edu/~brad/papers`

11. Efron, B., Tibshirani, R.J.: An Introduction to the Bootstrap. Chapman and Hall, New York (1993)
12. Ge, Y., Dudoit, S., Speed, T.: Resampling-based multiple testing for microarray data analysis. Test 12(1), 1–77 (2003)
13. Miller, R.G.: Simultaneous Statistical Inference, 2nd edn. Springer, New York (1981)
14. Newton, M.A., Noueiry, A., Sarkar, D., Ahlquist, P.: Detecting differential gene expression with a semiparametric hierarchical mixture method. Biostat. 5, 155–176 (2004)
15. Politis, D.N., Romano, J.P., Wolf, M.: Subsampling. Springer, New York (1999)
16. Romano, J.P., Wolf, M.: Exact and approximate stepdown methods for multiple hypothesis testing. Journal of the American Statistical Association 100, 94–108 (2004)
17. Shao, J., Wu, C.F.: A general theory of jackknife variance estimation. Ann. Statist. 17, 1176–1197 (1989)
18. Singh, D., Febbo, P.G., Ross, K., Jackson, D.G., Manola, J., Ladd, C., Tamayo, P., Renshaw, A.A., D'Amico, A.V., Richie, J.P., Lander, E.S., Loda, M., Kantoff, P.W., Golub, T.R., Sellers, W.R.: Gene expression correlates of clinical prostate cancer behavior. Cancer Cell 1(2), 203–209 (2002)
19. Tukey, J.W.: Bias and confidence in not quite large samples (Abstract). Ann. Math. Statist. 29, 614 (1958)
20. Tusher, V.G., Tibshirani, R., Chu, G.: Significance analysis of microarrays applied to the ionizing radiation response. Proc. Natl. Acad. Sci. U.S.A. 98, 5116–5121 (2001)
21. Westfall, P., Young, S.: Resampling-based multiple testing: Examples and methods for p-value adjustment. Wiley, New York (1993)

APPENDIX

The full rejection lists are given below, i.e., the gene (row) numbers declared non-null corresponding to the methods reported in Table 1.

Case thr= 0.2:

Data LIST: 2 11 298 332 341 364 377 579 610 702 805 905 914 1068 1077 1089 1113 1314 1346 1557 1588 1589 1620 1720 2370 3647 3665 3940 4331 4518 4546 4549 5158 5159

Subag (i) LIST: 2 11 35 73 212 292 298 332 341 364 377 423 452 478 518 579 594 610 611 637 642 660 684 692 694 698 702 709 718 721 731 735 739 758 805 813 905 914 921 987 1003 1018 1019 1068 1077 1082 1089 1090 1097 1113 1130 1254 1314 1329 1345 1346 1458 1476 1507 1557 1572 1588 1589 1620 1628 1659 1720 1966 2370 2385 2391 2852 2856 2912 2945 2968 3200 3269 3282 3375 3585 3600 3647 3665 3746 3930 3940 3991 4013 4040 4088 4104 4154 4163 4282 4316 4331 4396 4492 4496 4500 4515 4518 4546 4549 4552 4554 4671 4981 5158 5159 5305 5697

Subag (ii*) LIST: 2 11 35 73 298 332 341 364 377 423 452 478 518 579 594 610 611 637 642 660 684 692 694 698 702 709 718 721 731 735 739 758 805 813 905 914 921 1018 1068 1077 1089 1090 1113 1130 1254 1314 1329 1345 1346 1458 1476 1507 1557 1588 1589 1620 1628 1720 1966 2370 2385 2391 2852 2856 2912 2945 2968 3200 3269

3375 3585 3600 3647 3665 3746 3930 3940 3991 4013 4040 4088 4154 4163 4282 4316 4331 4396 4492 4496 4500 4515 4518 4546 4549 4552 4671 4981 5158 5159 5305 5697

Case thr= 0.3:

Data LIST: 2 11 298 332 341 364 377 579 610 611 637 702 735 805 813 905 914 1068 1077 1089 1113 1130 1314 1345 1346 1458 1507 1557 1588 1589 1620 1628 1720 2370 2856 2912 3647 3665 3940 3991 4088 4316 4331 4396 4492 4515 4518 4546 4549 5158 5159

Subag (i) LIST: 2 11 35 44 73 78 212 249 263 270 292 298 332 341 364 377 423 452 478 493 518 579 594 610 611 626 637 642 660 684 692 694 698 702 709 718 721 731 735 739 742 758 805 813 832 844 905 913 914 921 987 1003 1018 1019 1068 1077 1082 1089 1090 1097 1113 1130 1132 1254 1314 1329 1345 1346 1362 1458 1476 1491 1507 1508 1557 1566 1572 1588 1589 1620 1628 1643 1659 1702 1720 1872 1966 2370 2385 2391 2785 2852 2856 2912 2945 2968 3200 3208 3260 3269 3282 3375 3585 3600 3647 3665 3746 3930 3940 3961 3991 4013 4040 4057 4073 4088 4104 4154 4163 4282 4316 4331 4386 4396 4492 4496 4500 4510 4515 4518 4546 4549 4552 4554 4671 4981 5158 5159 5305 5547 5647 5697

Subag (ii*) LIST: 2 11 35 73 212 292 298 332 341 364 377 423 452 478 493 518 579 594 610 611 637 642 660 684 692 694 698 702 709 718 721 731 735 739 742 758 805 813 844 905 913 914 921 987 1003 1018 1019 1068 1077 1082 1089 1090 1097 1113 1130 1254 1314 1329 1345 1346 1362 1458 1476 1507 1508 1557 1588 1589 1620 1628 1659 1720 1966 2370 2385 2391 2852 2856 2912 2945 2968 3200 3208 3260 3269 3282 3375 3585 3600 3647 3665 3746 3930 3940 3991 4013 4040 4073 4088 4104 4154 4163 4282 4316 4331 4396 4492 4496 4500 4515 4518 4546 4549 4552 4554 4671 4981 5158 5159 5305 5547 5647 5697

Human Blood-Brain Differential Gene-Expression Correlates with Dipeptide Frequency of Gene Products

Shandar Ahmad

7-6-8, Saito-Asagi, Ibaraki-shi, Osaka 5670085, Japan
shandar@nibio.go.jp

Abstract. Differential gene expression in different tissues is largely considered to be the specific property of individual genes. In this work a relationship between overall dipeptide composition of proteins encoded by genes on the one hand and the difference in their expression level in two of the most important human organs i. e. blood and brain have been studied. Study is designed by developing a neural network that tries to predict the difference between expression of a gene in blood and brain from a 400-dimensional relative dipeptide composition vector. These vectors are derived from the amino acid sequence obtained by translating the corresponding gene. In a holdout validation scheme, such a model can predict gene expression from dipeptide composition with a significant Pearson's correlation of 0.49 with a classification capacity between (expression wise) blood favored and brain favored genes reaching 68 to 70% accuracy. Results indicate that despite diverse biological function of each expressed gene within a tissue, some similarities in gene products do exist.

1 Introduction

Gene expression is one of the most vigorously pursued area of research in Bioinformatics today. There has been a keen interest in understanding the mechanism of gene expression and for that purpose DNA-sequences, expressing in different environments have been studied. Prediction of gene expression from DNA-sequence attributes has been attempted recently [1]. Thus, it is important to know what determines gene expression at a chemical and physical level. In this work, an analysis of dipeptide composition of gene products i. e. proteins coded by expressed genes has been carried out. To determine if these features contribute to gene expression, a non-redundant set of 3761genes with known expression levels in human blood and brain have been analyzed by developing a neural network that would predict the difference in their expression levels in the two tissues. Test data sets are kept aside from training to assess true predictive role of dipeptide composition. Results on independent test sets show that about 68% genes can be correctly classified between those favoring to express in blood over brain or vice versa.

I. Măndoiu, R. Sunderraman, and A. Zelikovsky (Eds.): ISBRA 2008, LNBI 4983, pp. 504–508, 2008.

2 Methods

2.1 Data Sets

A dataset of human gene expression in different tissues has been compiled by Haverty et al. [2]. This so called HugeIndex database contains expression levels of genes in various tissues derived from human organs. Blood and brain are two of those most important organs, which have been used to carry out this study. Protein sequences encoded by these genes were obtained from EBI web server using their EMBLFETCH tool (http://www.ebi.ac.uk/cgi-bin/emblfetch?..). First a list of successfully completed queries was obtained. It was observed that some of the genes encoded for proteins with some sequence similarity. To avoid any bias caused by this, clustering of protein sequences was carried out using BLASTCLUST [8] to finally obtain a set of 3761 genes, such that no two members of the final data set were similar by more than 25%.

2.2 Differential Gene Expression

Gene expression level of a gene in brain was subtracted from that in blood and the difference of expression levels was normalized by a sigmoidal function as follows:

$$y(i) = 1/[1+\exp(x1(i) - x2(i))] \quad (1)$$

where x1(i) and x2(i) are the expression levels of gene i, in brain and blood respectively. This leaves behind a distribution of differential expression values between 0 and 1. A value close to 0 indicates higher expression in brain and closer to 1 indicates the same for blood. At y(i)=0.5, the two genes are expected to be equally expressed in the two systems.

2.3 Relative Amino Acid Composition

Using 20 amino acids, there are 20x20=400 possible dipeptide in a protein. Dipeptide amino-acid composition was calculated by using an obvious expression as follows:

$$d(ij) = N(ij)/N \quad (2)$$

where d(ij) is the relative frequency of occurrence of a dipeptide type j, in gene i and N(ij) is its absolute number. N is the total number of dipeptides encoded by the gene. Terminal positions from the sequence were not included.

2.4 Neural Network

Dipeptide composition of each gene product has been used as the inputs to a neural network. It is defined as follows:

$$D(i) = d(ij)\{j=1,..400\} \quad (3)$$

The target function is the expression level given by y(i) in (1). A hidden layer with two nodes was used to incorporate any non-linear or cumulative effect of input vectors. Number of training parameters would increase rapidly with the addition of every node in the hidden layer and therefore the number was kept at 2. To further limit the number

of trainable parameters, neural network biases in the nodes were turned off and set to zero for the entrie training and testing process. Detailed scheme of a neural network inputs and transformations of the training data has been shown in Figure 1.

The neural network was designed and trained using SNNS software [3], in quite a way similar to our other applications of neural networks (e.g. [4-5]).

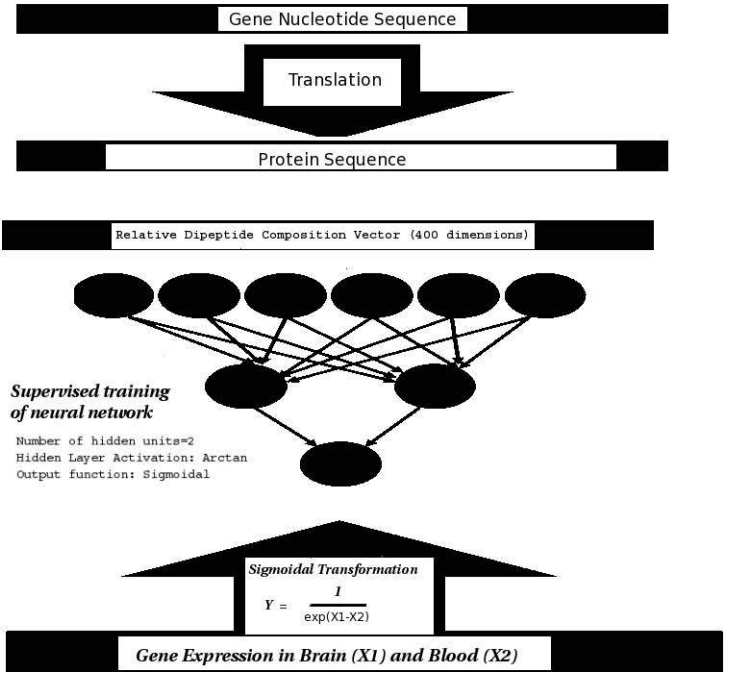


Fig. 1. Prediction scheme to estimate the relationship between differential gene expression and dipeptide composition of gene products

2.5 Cross Validation

There are less than 4000 genes and about 800 parameters to train and there is always a risk of over fitting in this system. To avoid this, the whole data was divided into five parts. In one training cycle, data from four of these parts is combined for training for a fixed number of training epochs (no information was used from the fifth left out data set, even to stop training, which is allowed to proceed for a maximum number of 100 cycles irrespective of over fitting the training data). Once the training is finished neural network is tested on the left out data and performance scores for this data are considered to be free from over fitting and are the ones being reported. Thus five cycles of training were performed by selecting each of the five parts of the data sets for evaluating true performance.

2.6 Performance Evaluation

Performance of a neural network is evaluated using standard scores such as Pearsons's correlation coefficient, sensitivity and specificity and net prediction (average of sensitivity and specificity, which is a good estimate of the area under the ROC curve). Net prediction (NP) was found to approximate the total number of correctly classified genes relative to the total number of classified genes (called accuracy). These scores have been repeatedly defined in literature [e.g. 4-5] and are not reproduced here assuming that they are obvious.

3 Results and Discussion

As stated in methods, five independent neural networks were trained to extensively examine the real predictive value of dipeptide compositions. Five pairs of training and test data sets were prepared and in each case training data was used to train the neural network, allowing over fitting but not letting any information from the test data enter the neural network. Neural network performance on the test data sets is finally reported in Table 1. As seen neural network could identify about 67 to 70% genes correctly to be over expressed in brain compared to blood or vice versa. It may be noted that the data contains many examples in which genes are not expressed at all in either of the two tissues and if we were to exclude them performance would improve. However, in this work all data was used so that all possible information could be extracted from composition vectors. Further, sensitivity is seen to be higher in the table, but the balance between sensitivity can be easily adjusted because the neural network actually returns a

Table 1. Neural network performance on test data sets over five independent cycles

Data	***Correlation***	***Sensitivity (%)***	***Specificity (%)***	***Net Prediction (%)***	***Accuracy (% correct prediction)***
Data1	0.46	67%	73%	70%	70%
Data2	0.47	96%	36%	67%	66%
Data3	0.49	91%	48%	70%	67%
Data4	0.52	85%	48%	66%	67%
Data5	0.49	62%	72%	67%	67%
Average	0.49	80%	55%	68%	68%

real value, which can be transformed into two-class predictions at different thresholds. Thus a balance between sensitivity and specificity can be manually selected with the constraint imposed by the maximum net prediction value.

Differential gene expression between two organs or tissues has not been studied in the current context before. However, some studies trying to predict gene expression from DNA sequence have used a large number of descriptors and also obtained a somewhat better prediction than the performance reported here [1,6,7]. Role of dipeptide composition in predicting overall gene expression has also been demonstrated [7]. It is expected that the use of more descriptors to estimate gene expression from sequence alone will further improve the prediction of tissue-specific gene expression. However, the goal of this work is to elucidate the extent to which dipeptide amio acid composition can estimate gene expression levels in two key human tissues. Work is in progress to achieve the best possible predictions using sequence-derived descriptors as well as to expand current analysis to the whole set of tissues for which data is available.

4 Conclusion

Tissue-specific gene expression has been analyzed by developing a neural network based prediction system. It is shown that the genes expressing in blood and brain do differ in terms of dipeptide composition of their translated proteins. Further improvements in prediction using additional descriptors of sequence as well as extension to other expression data are the works in progress.

References

[1] Beer, M.A., Tavazoie, S.: Predicting gene expression from sequence. Cell 117(2), 185–198 (2004)

[2] Haverty, P.M., Weng, Z., Best, N.L., Auerbach, K.R., Hsiao, L.L., Jensen, R.V., Gullans, S.: HugeIndex: A database with visualization tools for high-density oligonucleotide array data from normal human tissues. Nucleic Acids Res 30(1), 214–217 (2002)

[3] SNNS: Stuttgart Neural Network Simulator User manual, version 4.2, `http://www-ra.informatik.uni-tuebingen.de/SNNS/`

[4] Ahmad, S., Gromiha, M.M., Sarai, A.: Analysis and prediction of DNA-binding proteins and their binding residues based on composition, sequence and structural information. Bioinformatics 20, 477–486 (2004)

[5] Malik, A., Ahmad, S.: Sequence and structural features of carbohydrate binding in proteins and assessment of predictability using a neural network. BMC Struct Biol. 7, 1 (2007)

[6] Sharabiani, M.T., Siermala, M., Lehtinen, T.O., Vihinen, M.: Dynamic covariation between gene expression and proteome characteristics. BMC Bioinformatics 6, 215 (2005)

[7] Raghava, G.P., Han, J.: Correlation and prediction of gene expression level from amino acid and dipeptide composition of its protein. BMC Bioinformatics 6, 59 (2005)

[8] Altschul, S.F., Gish, W., Miller, W., Myers, E.W., Lipman, D.J.: Basic Local Alignment Search Tool. J. Mol. Biol. 215, 403–410 (1990)

Author Index